VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Autentizace směrovací informace v protokolu BGP

BGP Routing Information Authentication

2013 **Bc. Michal Gembík**

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Michal Gembík**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: Autentizace směrovací informace v protokolu BGP
BGP Routing Information Authentication

Zásady pro vypracování:

1. Prozkoumejte a zdokumentujte stav standardizace a implementace infrastruktury RPKI pro autorizaci propagace síťových prefixů ze zdrojových AS.
2. Vyhledejte nástroje pro ověření pravosti směrovací informace a implementace směrovacích prvků zohledňující ověření ve výběrových kritériích protokolu BGP (ROA) a ověřte je.
3. Prostudujte a zdokumentujte technologii SecureBGP (SBGP) pro autentizaci směrovací informace předávané mezi sousedy participující v BGP relaci a vyhledejte, ověřte a. srovnejte její implementace.
4. Zhodnoťte zvýšení zátěže vybraných implementací směrovačů při zavedení ověřování autenticity směrovací informace (proveďte praktická měření).

Seznam doporučené odborné literatury:

[1] Beijnum, I: BGP. O'Reilly Media, Inc, Sebastopol, CA 2002, ISBN 978-0-596-00254-1.
[2] Huston, G., Michaelson, G.: Validation of Route Origination Using the Resource Certificate Public Key Infrastructure (PKI) and Route Origin Authorizations (ROAs). RFC 6483, Internet Engineering Task Force (IETF), 2012, ISSN: 2070-1721.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Ing. Petr Grygárek, Ph.D.**

Datum zadání: 16.11.2012
Datum odevzdání: 07.05.2013

VYSOKÁ ŠKOLA BÁŇSKÁ-TECHNICKÁ UNIVERZITA * Fakulta elektrotechniky a informatiky * OSTRAVA *

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě dne 29.7.2013 .....

Rád bych poděkoval Ing. Petru Grygárkovi, PhD. za vedení diplomové práce, velice vstřícný přístup a odborné rady. Dále všem přátelům a kolegům za jejich podporu. A v neposlední řadě také mé rodině, za jejich podporu při studiu.

Seznam zkratek

RPKI - Resource Public Key Infrastructure

DER - Distinguished Encoding Rules

PKI - Public Key Infrastructure

CA - certificate Authority, certifikační autorita

CMS - Cryptographic Message Syntax

CRL - Certificate Revocation List

RIR - Regional Internet Registry

NIR - National Internet Registry

LIR - Local Internet Registry

ISP - Internet Service Provider

TAL - Trust Anchor Locator

TA - Trust Anchor

BGP - Border Gateway Protocol

EE - End Entity certifikát

IANA - Internet Assigned Numbers Authority

SIA - Subject Information Access

PDU - Protocol Data Unit

SIDR - Secure Inter-Domain Routing

IETF - The Internet Engineering Task Force

X.509 - Standard, který definuje formát a syntaxi certifikátů

SHA-x - (Secure Hash Algorithm) skupina typů hashovacích funkcí

RP - Relying party

ROA - Route Origin Authorizations

URI - Uniform Resource Identifier

INR - Internet Number Resources

API - Application Programming Interface

Abstrakt:

Diplomová práce se zabývá prozkoumáním a zdokumentováním „problematiky“ standardizace a implementace infrastruktury Secure BGP a RPKI pro autorizaci síťových prefixů ze zdrojových autonomních systémů. Tato standardizace si klade za cíl zlepšit zabezpečení BGP infrastruktury pomocí certifikace přidělených čísel autonomních systémů a rozsahů IP adres jednotlivým držitelům těchto zdrojů. Další dílčí částí této diplomové práce je ověření a nasazení vyhledaných nástrojů pro implementaci infrastruktury RPKI, které jsou dále porovnány z hlediska rychlosti validace jednotlivých sítí.

Klíčová slova: RPKI, Validátor, Certifikační autorita, secure BGP, ROA, BGP

Abstract:

This thesis deals with analyze and documenting problem of standardization and implementation SecureBGP and RPKI infrastructure for authorization of network prefixes from source autonomous systems. Aim of this standardization is improve security of BGP infrastructure by using certification of autonomous system numbers and ranges of IP addresses allocated to each holder these resources. Other part of this thesis is verified and deployment of matching tools for implementing RPKI infrastructure and then these tools are compared in terms of speed validation of networks prefixes.

Key words: RPKI, Validator, Cerificate authority, secure BGP, ROA, BGP

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod

Stále se rozšiřující BGP infrastruktura s rostoucím počtem BGP cest vedla v minulosti ke stále většímu počtu problémů a bezpečnostních rizik. Jelikož všechny internetové přenosy z libovolné domény prochází přes hraniční BGP směrovače, které jsou hlavními tranzitními body pro jednotlivé autonomní systémy nebo sítě, je zde potřeba bezpečnostních opatření. Jedním z rizik je, že každý BGP směrovač může jak úmyslně tak i neúmyslně propagovat nesprávné informace o dosažitelnosti jednotlivých sítí, které se rozšíří do zbytku celé BGP infrastruktury. Těmito informacemi mohou být čísla AS a rozsahů IP adres přidělená konkrétním subjektům, které jsou oprávněny tyto zdroje využívat.

Chyby způsobené špatnou propagací cest mají za následek, jak nestabilitu sítě, tak také i samotnou dosažitelnost dané sítě. Známým problémem kdy v roce 2008 byla způsobená nedosažitelnost serveru YouTube, kdy informace určené serveru YouTube byly přesměrovány do sítě PakistanTelecom [http://www.renesys.com/2008/02/pakistan-hijacks-youtube-1/]. V roce 2010 se objevil další problémem, kdy v ChinaTelecom došlo k nesprávnému oznámení a následné propagaci cest, které se dotklo více než 50000 různých bloků IP adres, a tyto informace se rozšířily do více než stovky různých zemí celého světa. [http://www.renesys.com/2010/11/chinas-18-minute-mystery/]

První část této práce je věnována popisu řešení zabezpečení pomocí Secure-BGP (SBGP) a secure origin BGP (soBGP). Jelikož k této problematice nebyly nalezeny dostupné implementace, tak zbytek práce je věnováno hlavnímu tématu práce, kterým je „Resource Public Key Infrastructure“ (RPKI) a je nástupcem těchno starších řešení. V druhé části této práce je popsána samotná infrastruktura RPKI, která se na rozdíl od S-BGP a soBGP v dnešní době stále vyvíjí.

Praktická část této práce je věnována samotnému nasazení a ověření vyhledaných implementací RPKI pomocí kterých bylo možné tuto problematiku ověřit, otestovat a srovnat jednotlivá nalezená řešení.

2 Secure BGP

Jednou z hlavních myšlenek pro dosažení zabezpečení BGP infrastruktury bylo navrhnout prostředky k ověření základních tvrzení týkajících se původu trasy do inter-domain směrovacího systému. Dalším krokem pak bylo ověřit, že používané bloky IP adres a čísla autonomních systémů (AS) jsou platné a jednotlivé subjekty jsou pro používání těchto zdrojů oprávněny. Pro vývoj metody, jak dosáhnout zabezpečení, byla jako první kolem roku 1997 pověřena americká společnost BBN. V následujících deseti letech je pak publikováno mnoho dokumentů, vytvořených různými společnostmi, obsahující různé formy zabezpečení protokolu BGP. Tyto dokumenty popisují konkrétní formy, kterými jsou Secure-BGP (S-BGP) [1][2], Secure Origin BGP (soBGP) [3][4] a Pretty Secure BGP (psBGP) [5]. V následujících podkapitolách se podíváme na řešení zabezpečení BGP infrastruktury pomocí S-BGP a soBGP.

2.1 S-BGP

Tento protokol zajišťuje ověření IP adres a čísel AS obsažených v BGP zprávě UPDATE, které si mezi sebou vyměňují jednotlivé směrovače v BGP infrastruktuře, pomocí kterého pak můžeme zjistit jestli daný IP adresní prostor patří pod správu daného AS. S-BGP architektura pochází od společnosti BBN a využívá tři hlavní bezpečnostní mechanismy, kterými jsou Publik Key Infrastructure (PKI), Attestation a IPsec technologii.

2.1.1 Public key Infrastructure

Mechanismus PKI se používá pro ověřování vlastnictví a propagaci bloků IP adres a čísel AS. V infrastruktuře S-BGP je každý AS opatřen veřejným a soukromým klíčem. Smyslem S-BGP je opatřit každou odeslanou zprávu UPDATE podpisem pomocí soukromého klíče. Když příjemce tuto zprávu obdrží, může ji ověřit pomocí veřejného klíče. PKI tím pádem napomáhá k identifikaci všech zúčastněných uzlů včetně původního odesilatele.

2.1.2 Attestation

Jedná se o „optional transitive“ atribut použitý pro zapouzdření autorizační informace uvnitř zpráv UPDATE. Struktura zprávy UPDATE se zapouzdřeným atributem „attestation“ je ukázána na Obrázek **1**. Na tomto obrázku můžeme vidět strukturu attestation obsažená v „BGP Path Attribute“ ve kterých se nachází informace například o vydavateli certifikátu „Issuer“, dále pak informace o samotných digitálních podpisech,

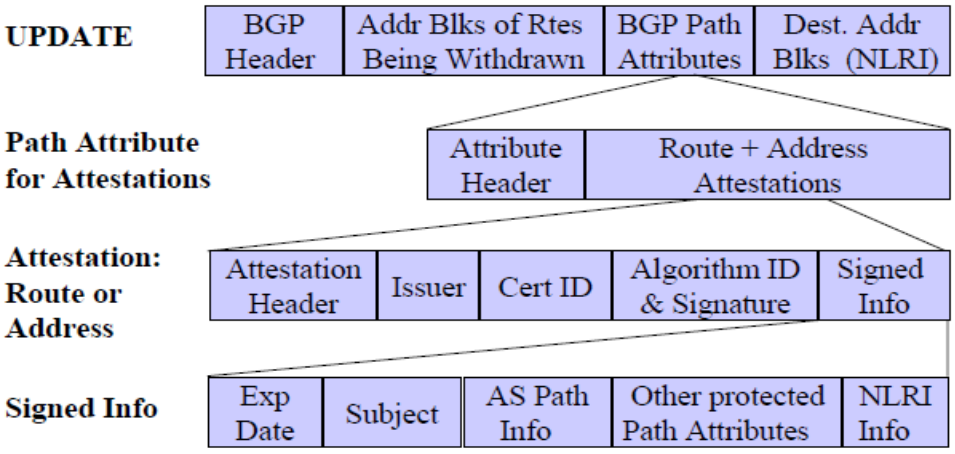
které byly použity pro podepsání ověření, že AS má právo inzerovat určitý adresní blok nebo je jeho vlastníkem. Tato informcace, pak zajišťuje pravost poskytovaných údajů.

Atribut attestation obsahuje posloupnost všech digitálních podpisů, které postupně podepisovali jednotlivé AS v rámci celé trasy aby zajistili důveryhodnost posílané informace (čísla AS a IP adres sítí). V případě, že má organizace ve správě více než jeden AS, pak pro každý AS musí existuje samostatný certifikát.

Existují dva typy osvědčení, které sdílí stejný formát atributu attestation. Těmito typy jsou:

Address Attestation - jsou vydány samotnou organizací vlastnící IP prefix obsažený ve zprávách UPDATE a opravňuje AS inzerovat cesty k zadaným blokům adresního prostoru.

Route Attestation - jsou postupně vydávány samotnými autonomními systémy a jsou použity k ověření dané cesty. V případě, že směrovač obdrží od souseda zprávu BGP update, a ověří že infromace v ní obsaženě jsou validní přidá k této zprávě svůj podpis.



Obrázek 1: Struktura zprávy UPDATE s atributem „Attestation“

2.1.3 IPsec

Z předchozího textu by se mohlo zdát, že Attestation obsažené v každé zprávě UPDATE, můžou zcela BGP infrastrukturu zabezpečit. Nechrání ji však před některými útoky, jako například DoS útoky. Tento problém je vyřešen pomocí IPsec bezpečnostních mechanismů na síťové vrstvě v OSI modelu, jako je „autentifikace“ a „kryptování“, které jsou použit právě pro zabezpečení BGP provozu mezi dvěma sousedními směrovači, a tím zajišťuje ověření integrity zpráv, totožnosti odesilatelů a skutečnosti, že zpráva je poslána správnému příjemci.

S-BGP komplexně pokrývá široké spektrum bezpečnostních problémů spojených s BGP. Detekuje a odmítá neoprávněné zprávy UPDATE, které mohou vzniknout například chybnou konfigurací. Má však také své nedostatky. Jedním z hlavních nedostatků je například skutečnost, že v průchodu přes jednotlivé směrovače zpráva UPDATE stále narůstá.

2.2 soBGP

Secure Origin BGP protokol byl navržen společností CISCO. Od roku 2002 do roku 2006 vyšla řada článků o soBGP, které tento protokol popisují (jedná se o draft dokumenty) Protokol soBGP poskytuje bezpečnostní mechanismus, pomocí kterého může BGP směrovač ověřovat AS_PATH obsažený ve zprávě UPDATE. soBGP, podobně jako S-BGP, ověřuje oprávnění AS uvedeného v každé zprávě UPDATE propagovat dosažitelnost daného prefixu a dále také ověřuje platnost AS_PATH.

Platný atribut **AS_PATH** je cesta, která má tyto vlastnosti:

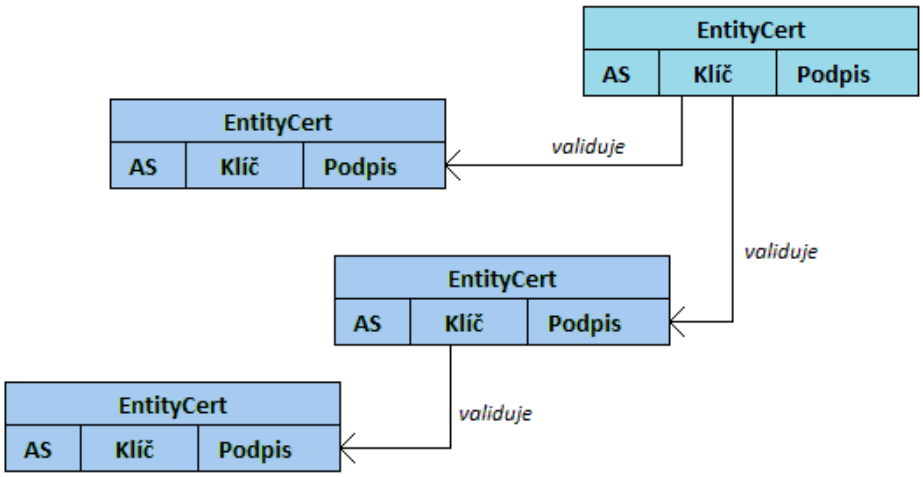
- Každý AS, který je uveden v AS_PATH je skutečným AS v cestě.
- Každé dva AS uvedené za sebou v atributu AS_PATH jsou skutečnými sousedy.
- Cílového AS je dosaženo opravdu průchodem od prvního AS uvedeného v AS_PATH přes každý následující uvedený AS.

Hlavním principem protokolu soBGP je za pomocí distribuční sady podepsaných certifikátů obsahující požadované informace sloužící k ověření výše zmiňovaného oprávnění daného AS propagovat dosažitelnost daného prefix. V případě, že jsou tyto certifikáty přijaty a zpracovány je vytvořena databáze která obsahuje:

- Bloky IP adres a čísla autonomních systémů oprávněných tyto bloky IP adres propagovat.
- Politiky aplikovatelné na určené bloky IP adres a jednotlivých prefixů.
- Seznam navzájem propojených AS v síti. Tento seznam je dále použit k vytvoření grafu navzájem připojených AS v síti viz. Obrázek **4**.

Z této databáze jsou vytvořeny registry směrovacích informací, které jsou používány k ověření platnosti směrovacích informací od jednotlivých BGP sousedů.

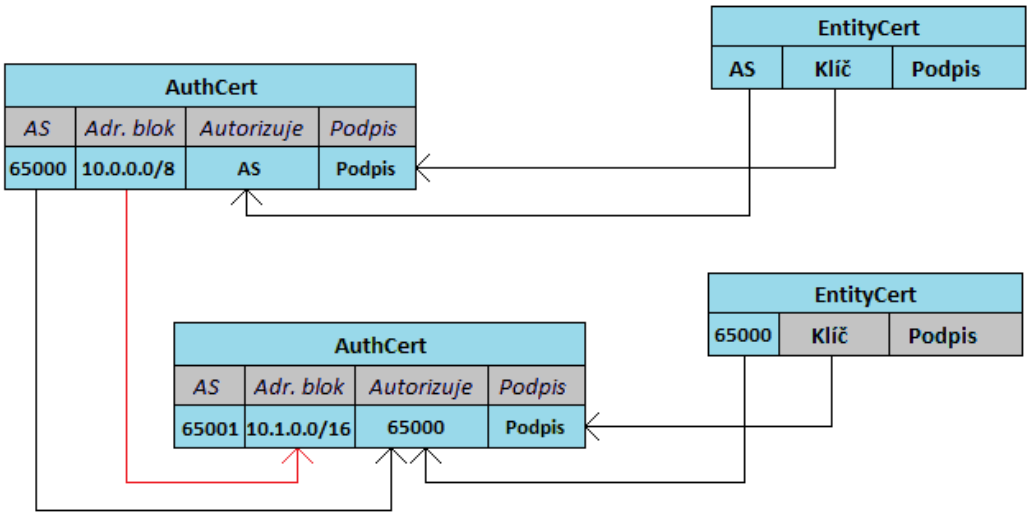
Důležitým bodem zabezpečení je mít ověřené sousedské vazby. SoBGP toho docílí pomocí Entity certifikátů, jejichž struktura je definována standardem X.509v3. Každý z těchto certifikátů váže číslo konkrétního AS ke zveřejněnému číslu alespoň jednoho dalšího AS. Tuto vazbu nám popisuje Obrázek **2**. Tyto certifikáty jsou ověřovány jednotlivými certifikačními autoritami nejvyšší úrovně, a proto jim důvěřuje každý autonomní systém. Autonomní systémy a jejich EntityCert certifikáty, pak můžete sami prohlásit za důvěryhodné podepsáním vlastním klíčem.



Obrázek 2: Znázornění důvěry mezi EntityCert.

V případě, že máme Entity certifikáty, které poskytují ověření pro každý AS, pak inzerování jednotlivých bloků IP adres danými AS je dalším problémem, který soBGP řeší, a to pomocí certifikátu AuthCert. Certifikát AuthCert se tedy používá pro spárování AS s daným IP adresním blokem, který jej opravňuje používat. Certifikát AuthCert je podepsán právě Entity certifikátem a tento vztah popisuje Obrázek **3**.

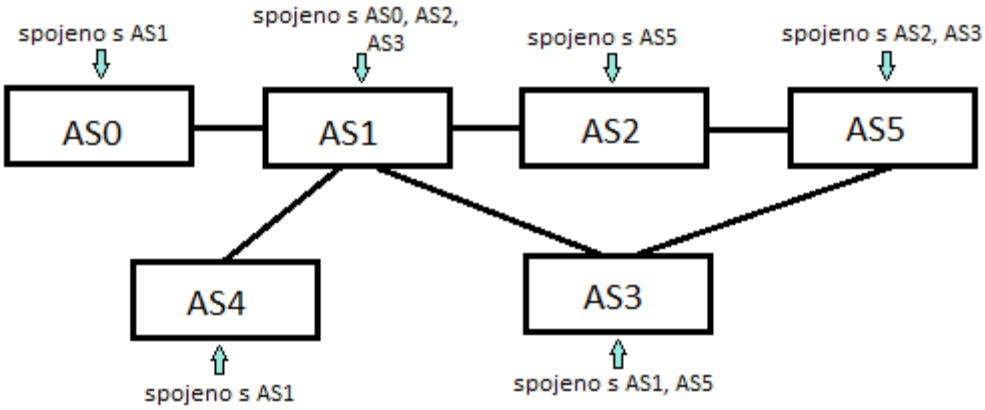
Jestliže chceme přenést část z alokovaného adresního bloku do dalšího AS, pak AS který vlastní tento adresní blok sestaví nový AuthCert, ve kterém je připojen určitý adresní blok jinímu AS.



Obrázek 3: Vztah mezi EntityCert a AuthCert.

Volitelnou součástí soBGP protokolu je možnost nastavení politiky vztahujících se k danému rozsahu IP adres. Tuto možnost nám dává certifikát PrefixPolicyCert (podrobný popis těchto certifikátů můžeme najít v literatuře [3] kapitola 5).

Další otázkou je, zda jednotlivé AS propagující danou trasu, skutečně mají cestu k cíli. Tuto otázku řeší topologická mapa cest navzájem propojených sítí. Příklad topologické mapy můžeme vidět na Obrázek **4**. Aby mohla být tato topologická mapa sestavena, musí každý AS vytvořit certifikát ASPolicyCert obsahující seznam sousedů připojených k danému AS, který musí být opatřen podpisem pomocí privátního klíče daného AS. Právě na základě seznamu jednotlivých sousedů může být sestavena topologická mapa.



Obrázek 4: Topologická mapa navzájem připojených AS

Pro přenos jednotlivých certifikátů (EntitiCert, PrefixPolicyCert, ASPolicyCert) mezi jednotlivými sousedními AS byla v soBGP navržena nová zpráva „SECURITY message“, která výměnu certifikátů zprostředkovává mezi jednotlivými soBGP sousedy. V případě, že dva sousední soBGP posluchači mají navázanou vazbu mohou si poslat právě pomocí zprávy „SECURITY message“ certifikáty obsažené v jejich lokálních databázích a tyto jsou pak zpracovávány samotnými směrovači. Druhou možností je, že samotné směrovače tyto certifikáty nezpracovávájí, ale přeposílají je na lokální servery pomocí kterých se jednotlivé certifikáty spracují a směrovače se pak dotazují jednotlivých serverů zda přijaté směrovací informace od sousedů jsou platné či nikoli.

2.3 Srovnání

Jak bylo popsáno výše, S-BGP a soBGP protokoly byly navrženy, aby vyřešily jistá bezpečnostní rizika. Oba tyto protokoly řeší tato rizika pomocí digitálních podpisů. S-BGP řeší zabezpečení pomocí dynamického podpisu každé zprávy UPDATE, která se přeposílá mezi BGP sousedy. Tím nabízí lepší zabezpečení než soBGP protokol. Na druhou stranu touto vlastností S-BGP z velké části zatěžuje infrastrukturu tím, že velikost zprávy UPDATE stále narůstá. To zvyšuje složitost daného problému a klade velké nároky na šířku pásma a výkon jednotlivých směrovačů, což může ovlivnit samotné zpracování celé zprávy UPDATE.

soBGP nezatěžuje celou infrastrukturu do takové míry, jako je tomu u S-BGP, jelikož samotné certifikáty mohou být podepisovány jinou entitou a jednotlivé bezpečnostní mechanismy jsou posílány pouze pomocí SECURITY zprávy a to jen mezi sousedními AS.
S rostoucími nároky na bezpečnost a vývoj celé BGP infrastruktury byly S-BGP a soBGP protokoly vytlačeny novým principem zabezpečení, který se nazývá RPKI. Z tohoto důvodu

nebylo možné najít žádné dostupné implementace, které by umožnily celou problematiku otestovat v laboratorním prostředí. Proto se těmito architekturami nebudeme již dále zabývat.

3 Resource Public Key Infrastructure

Resource Public Key Infrastructure (RPKI) je infrastruktura, která využívá digitálně podepsané certifikáty a objekty založené na standardu X.509 PKI [6], který byl rozšířen o IP adresy a čísla autonomních systémů [7]. Dále pak využívá podepsané objekty, kterými jsou „manifesty“ a „Route Origin Authorizations“ (ROA). Struktura těchto objektů je uvedena v RFC6488 dokumentu [8]. Podstata architektury RPKI je také uvedena detailně v RFC6480 dokumentu [9]. Cílem RPKI infrastruktury je zlepšit bezpečnost v oblasti BGP směrování a poskytovat ověřené údaje o tom, který AS je oprávněn propagovat vlastní přiřazené sítě do BGP infrastruktury. Informace z RPKI infrastruktury určitým způsobem dokáží zmírnit riziko napadení nejrůznějšími typy útoků na BGP infrastrukturu, jako jsou například neoprávněné propagace a užívání sítových adres a AS (Prefix Hijacks, Sub-prefix Hijacks), kterým například Pakistan Telecom 24. února 2008 způsobil zablokování serveru YouTube.

Definováním standardů pro RPKI se od roku 2007 začala zabývat IETF, která vytvořila pracovní skupinu SIDR. Ta se touto problematikou zabývá do současnosti. Počáteční fáze představování infrastruktury RPKI světu jednotlivými „Regional Internet Registry“ (RIR) začala až od roku 2009, kdy jednotliví RIR vydaly nezávislé certifikáty podepsané jejich privátním klíčem pro jimi spravované IP rozsahy a autonomní systémy. Jednotlivými RIR v dnešní době jsou RIPE NCC, AfriNIC, APNIC, ARIN, LACNIC.

IP adresní prostor, jak už bylo zmíněno výše, se obvykle přiděluje hierarchicky, kdy „IANA“ je kořenem hierarchie, který rozděluje IP adresní prostory a čísla AS jednotlivým RIR. RIR pak rozděluje podmnožiny svého adresního prostoru jednotlivým NIR nebo LIR které má pod svou správou. Tyto subjekty pak mohou dále rozdělovat svůj adresní prostor mezi jiné ISP nebo koncové zákazníky. Následujících podkapitoly slouží k vymezení pojmů v RPKI infrastruktuře ve kterých si popíšeme jednotlivé prvky objevujícíse v RPKI.

3.1 Certifikační autorita (CA)

CA můžeme chápat dvěma způsoby. Buď se může jednat o instituci, která zajišťuje proces vydávání certifikátů, anebo o aplikaci, která umožní vydávání certifikátů. V následujícím textu bude následně pro aplikaci certifikační autority použita zkratka CA.

3.1.1 CA certifikáty

Každý držitel jakéhokoli adresního bloku a zároveň čísel AS pro které jsou tyto bloky přiděleny je zároveň oprávněn tyto zdroje přidělovat a musí být schopný i vydávat certifikáty pro ověření přidělených zdrojů. Struktura certifikátu je popsána v RFC6487 [10]

RPKI certifikát musí obsahovat tyto pole:

Version - Verze certifikátu, která musí být v systému RPKI standardu X.509 ve verzi 3.

SerialNumber - Sériové číslo certifikátu, které musí být kladné a musí být jedinečné pro každý certifikát vydaný danou certifikační autoritou.

SignatureAlgorithm - typ algoritmu. V RPKI infrastruktuře se používají dva algoritmy. RSA pro certifikáty, seznam odvolaných certifikátů CRL a podepsané objekty (manifesty, ROA). SHA-256 slouží jako hashovací algoritmus. Podrobnější popis můžeme najít v dokumentu (RFC6485) [11]

Issuer - Certifikační autorita, která podepsala daný certifikát. Hodnota tohoto pole je platná podle standardu X.501. Toto pole musí obsahovat atribut CN (CommonName) a také může obsahovat jednu instanci atributu SerialNumber. Atribut CN musí být kódován pomocí ASN.1. RPKI se nespoléhá na názvy vystavovatele, ale přesto jsou z bezpečnostních důvodů globálně jedinečné, což minimalizuje pravděpodobnost kolize.

Subject - Jméno vlastníka veřejného klíče. Toto pole má stejná omezení jako pole Issuer. V RPKI název Subjektu určuje vydavatel (Issuer) a ne samotný Subjekt.

Validity - Jedná se o dobu platnosti certifikátu. Platnost certifikátu je reprezentována těmito poli:

NotBefore – Datum začátku platnosti.
NotAftrer – Datum skončení platnosti.

SubjectPublicKeyInfo - toto pole obsahuje informaci o veřejném klíči vlastníka, jaký algoritmus byl zvolen pro veřejný klíč a samotný veřejný klíč.

Resource Certificate Extensions - pole je použito například pro odkaz kde se nachází seznam odvolaných certifikátů. Dále pak obsahuje jednoznačný identifikátor certifikátu obsahující veřejný klíč vlastníka certifikátu. Další hodnotou je pak **AIA (Authority Information Access)** která obsahuje URI odkazující na autoritativní umístění pro CA certifikát, pod kterým byl daný certifikát vydán. Toto pole je volitelné pro RPKI. (RFC 6487 kapitola 4.8).

3.1.2 Seznam odvolaných certifikátů

Obsahem seznamu odvolaných certifikátů (CRL) jsou certifikáty, které byly zrušeny certifikační autoritou, a stále jim nevypršela doba platnosti. Podrobný popis CRL pro RPKI je uvedeno v RFC5280 [6]

V RPKI je potřeba vydávat tyto certifikáty ve verzi 2, protože spoléhající se strany nejsou povinné zpracovávat CRL verze 1. Vystavovatelem je samotná certifikační autorita, a pokud je vydáno více CRL jednou certifikační autoritou, tak CRL s nejvyšším číslem nahrazuje všechny ostatní CRL vydané touto certifikační autoritou.

V každém CRL musí být rozšiřující pole, které obsahuje veřejný klíč shodujícíse s priváním klíčem použitým k podepsání daného CRL a číslo seznamu CRL. Dále pak spoléhající se strana musí být připravena zpracovat tyto rozšíření.

V seznamu odvolaných certifikátů musí být pro každý odvolaný certifikát pouze dvě pole identifikující sériové číslo a datum odvolání certifikátu. Ostatní pole v seznamu odvolání nesmí být zahrnuty. Certifikační autorita může odvolání nebo zneplatnění prefixů v RPKI infrastruktuře provést následujícími způsoby:

- Přepsáním certifikátu nově vydaným certifikátem s menší nebo jinou sadou IP zdrojů.
- Zrušením a smazáním vydaných certifikátů z RPKI uložiště.
- Vytvořením nového certifikátu.
- Zamítnutí vydání nového certifikátu po jeho expirační době.

3.1.3 Trust Anchor a Trust Anchor Locator

Trust anchor (TA), nazývaný také jako bod důvěry, je reprezentován Self-Signed certifikátem. Tento typ certifikátu je podepsaný samotnou certifikační autoritou a musí být platný podle standardu X.509 ve verzi 3. Vzhledem k tomu, že TA je Self-Signed certifikát, tak mu neodpovídá žádný CRL, který by mohl být použit k zneplatnění certifikátu, ani žádný manifest (seznam všech objektů v repositáři, viz. kapitola 3.5), který by tento certifikát obsahoval.

Trust Anchor Locator (TAL) slouží RPKI validátoru tím, že odkazuje na repositář CA (3.1.7) a samotný veřejný klíč, který je použit k ověření, že certifikát v repositáři patří opravdu dané autoritě. Na základě toho pak validátor může stáhnout a ověřit další certifikáty a také podepsané objekty, kterými mohou být manifesty nebo Route Origin Authorization (ROA) (viz. kapitola 3.7) v rámci daného repositáře. Podrobnou definici TAL můžeme najít v RFC6490 [12]

TAL je uspořádanou posloupností:

- Rsync URI
- <CRLF> nebo <LF> konec řádku
- Informace o veřejném klíči vlastníka v DER formátu zakódovaného pomocí Base64 kódování

Příklad:

```
rsync://localhost/root/root.cer

MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAxnFmfJA203g0H3PgR44C
Mh3c644IEdgxEibPaD01UsEerEBH9L7Dvwy8O2SaLr663fl+/EhM2naCeOjFXi4h
eChM0/j8eQ6LSEl4KgFH50EVQgst8x3qUbA1oXq3+L9cJyfAi2K8qqjywACEOEJB
XLwQg0PR5ktifjENp+TZ2pIptdpl+BNWfT3e8KwfRs0kaCFek/iWg6pKitdqIRcZ
9fbmc+8+wTpuNATDmtDXQlKds71JcVHRwCe37ig4NMLV+d8xtFWfxCopo09NeBIN
98onousPfQUwnA1LuJYeAR6NRd+dV/UNSiKXLWLTnMO5AFNI0kTYl4VOw7pVCpr9
XwIDAQAB
```

TAL a TA certifikát musí splňovat tyto požadavky:

- Rsync URI musí odkazovat na právě jeden konkrétní objekt (nesmí však odkazovat na adresář ani soubor objektů). Daný objekt, pak musí být Self-Signed certifikát vydaný danou certifikační autoritou a musí odpovídat profilu RPKI certifikátu.
- Délka platnosti TA certifikátu by měla reflektovat očekávané období stability pro konkrétní sadu Internet Number Resources (INR), které jsou sdruženy s daným TA.
- V TA certifikátu se veřejný klíč nesmí po dobu jeho platnosti změnit.
- V případě, že z jakéhokoli důvodu je TA certifikát obnovený, musí být tento certifikát zveřejněn na stejném URI.

3.1.4 Manifest

Manifest je v infrastruktuře RPKI digitálně podepsaným objektem (specifikovaný v RFC6486 dokumentu) [13], který obsahuje seznam všech objektů v repositáři dané certifikační autority, která je odpovědná za tento zveřejněný obsah.

Manifest obsahuje tyto informace:

- seznam platných certifikátů
- nejnovější CRL vydaný touto certifikační autoritou.
- všechny vydané podepsané objekty, které jsou ověřitelné pomocí End-Entity (EE) certifikáty vydaných touto certifikační autoritou.

3.1.5 End-Entity (EE) Certifikáty

Primárním účelem těchto certifikátů je ověřování podepsaných objektů, které se vztahují k používání zdrojů IP adresních bloků a čísel AS, jako například ROA a manifesty. Z tohoto důvodu se tyto certifikáty nepoužívají k podepisování ostatních certifikátů v RPKI.

Mezi End-Entity certifikátem a jím podepsaným objektem je vztah jeden k jednomu, což znamená, že privátní klíč korespondující s každým End-Entity certifikátem je použit k podepsání právě jednoho objektu a každý objekt je podepsán právě jedním klíčem. Když je EE certifikát zneplatněn, je zneplatněn i objekt ním podepsaný. Toto umožnuje RPKI zneplatnit objekty bez potřeby existence dalšího speciálního mechanizmu.

Další výhodou tohoto vztahu je, že privátní klíč EE certifikátu je použit pouze jednou v období existence daného certifikátu, a to při podpisu objektu. Po podpisu objektu může být klíč zničen, což může zjednodušit správu klíčů, jelikož není potřeba, aby byly tyto privátní klíče uchovávány po dlouhou dobu. EE certifikát je součástí Cryptographic Message Syntax (CMS), takže není nutné jej posílat odděleně od podepsaného objektu, stejně jako není potřebné, aby byl EE certifikát zobrazen v RPKI repositáři samostatně.

EE certifikát může být použit k ověření sekvence podepsaných objektů, za předpokladu že, každý podepsaný objekt přepíše v repositáři předchozí výskyt dříve podepsaného objektu. To znamená, že je publikovaná v jednom čase jen jedna instance podepsaného objektu. Tyto certifikáty jsou proto nazývany také jako „sequential use“.

EE certifikáty používané k ověření pouze jedné instance Signed objektu, které nejsou použity pro další ověřování, se nazývají „One-time-use “ certifikáty.

3.1.6 Route Origin Authorization

Route Origin Authorization (ROA) je kryptograficky podepsaný objekt, který obsahuje seznam prefixů a číslo autonomního systému, a který využívá šablonu pro digitálně podepsané objekty RPKI [8]. Tato šablona definuje Cryptographic Message Syntax (CMS) (specifikována RFC5852 dokumentem) a můžeme ji chápat jako obal pro ROA, obsah a rovněž obecné validační procedury pro RPKI podepsané objekty. CMS obal obsahuje End-Entity (EE) certifikát pro klíč použitý k ověření ROA. Samotný nadřazený EE certifikát pak musí ve svém rozšíření pro RPKI certifikáty obsahovat stejné IP prefixy, jaké jsou uvedené v ROA.

Z předchozího plyne, že ROA definuje vztah mezi AS a bloky IP adres. Toto určuje, jaký AS je oprávněn užívat určitý IP prefix nebo množinu IP prefixů. Samotný podrobný profil pro ROA je popsán pomocí dokumentu RFC6482 [14].

ROA se tedy skládá ze tří prvků:

- číslo autonomního systému
- prefix vlastněný daným autonomním systémem
- maximální délka prefixu

ROA
AS 200
10.0.0.0/8-16

Maximální délka prefixu určuje nejkonkrétnější IP prefix, který je autonomní systém oprávněn propagovat. Pokud hodnota maximální délky prefixu není nastavena, pak autonomní systém je oprávněn inzerovat jen právě tento specifikovaný prefix. Kterékoli specifičtější IP prefixy budou považovány za neoprávněné. Toto je způsob, jak zabránit únosu přes vyhlášení konkrétnějšího prefixu.

Oznámení platnosti ROA

V případě, že chceme v RPKI infrastruktuře ověřit platnost BGP cesty (vztah mezi AS a IP adresou sítě), musíme vytvořit ROA objekt pro kombinaci čísla AS a IP prefixu, kterým ovlivníme chování směrovače pro konkrétní propagovanou BGP cestu RFC6483 [15].

V následující části si představíme, jakých hodnot mohou nabývat prohlášení o propagovaných cestách. U každé varianty můžeme vidět jednotlivé příklady.

Prohlášení o propagovaných cestách:

VALID – prohlášení znamená, že cesta je validátorem prohlášena za platnou, pokud existuje ROA obsahující stejnou IP adresu sítě, stejný prefix sítě a stejné číslo AS, jako obsahuje samotná cesta uvedena v BGP updatu. Také když BGP prefix v BGP updatu spadá do maximální délky uvedené v ROA.

Příklady:

ROA 10.3.0.0/20 AS8
BGP update : 10.3.0.0/20 AS8

ROA 10.3.0.0/20-22 AS8
BGP update: 10.3.0.0/21 AS8

INVALID – prohlášení které určuje, že pro cestu byl nalezen alespoň jeden objekt ROA odpovídající IP adrese sítě a jejímu prefixu (nebo spadající do maximální délky prefixu), ale neodpovídající číslu AS.

Příklad:

ROA 10.3.0.0/20 AS8
BGP update: 10.3.0.0/21 AS5

V opačném případě, jako je uvedeno u předchozího příkladu, daná cesta odpovídá číslu AS, ale neodpovídá prefixu sítě nebo maximální délce prefixu a zároveň je daný prefix specifičtější než prefix obsažený v ROA.

Příklad:

ROA 10.3.0.0/20-24 AS8
BGP update: 10.3.3.0/26 AS8

V případě, že BGP update má specifičtější prefix s jiným AS pak je tato cesta také prohlášena za **INVALID**.

Příklad:

ROA 10.0.0.0/16-24 AS1
BGP update: 10.0.0.0/32 AS2

UNKNOWN (NotFound) - prohlášení nám určuje, že pro danou cestu nebyl nalezen žádný objekt ROA odpovídající dané síti s daným prefixem ani daným číslem AS

Příklad:

ROA - žádné
BGP update: 10.3.0.0/8 AS13

Pokud ROA obsahuje specifičtější prefix nebo délku prefixu než je uvedené v BGP update a číslo AS je stejné nebo různé, pak daná cesta je prohlášena za „NotFound“

Příklady:

ROA 10.3.0.0/16-24 AS 8
BGP update: 10.3.0.0/8 AS8

ROA 10.3.0.0/16-24 AS 8
BGP update: 10.3.0.0/8 AS9

3.1.7 Uložiště certifikátů

Jednotlivé certifikáty vydané CA jsou umístěny v publikovaném uložišti (repositáři), které je obvykle reprezentováno jako veřejně přístupný adresář souborového systému [16]. Tento adresář je dostupný pomocí URI. Pro každý certifikát a podepsaný objekt dané CA v RPKI infrastruktuře, odpovídá jedno publikované uložiště, které je autoritativním publikačním bodem pro konkrétní CA.

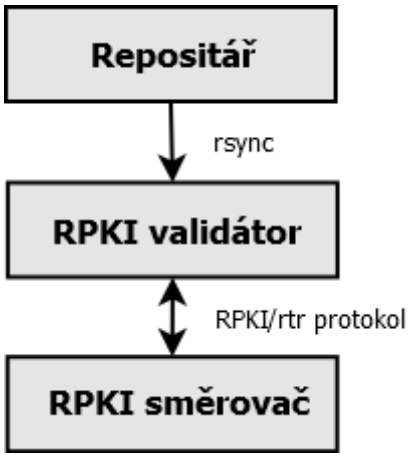
RPKI uložiště certifikátu můžeme chápat jako systém skládající se z několika instancí jednotlivých uložišť, kde každá instance se skládá z jednoho nebo více uložišť. Každé publikované uložiště je použito jedním nebo větším počtem subjektů uvedených v RPKI certifikátech, které jsou definovány v Subject Information Access (SIA). V hodnotě SIA

můžeme najít URI odkazující na uložiště, kde je samotný certifikát vystaven a také nastavené přístupové mechanizmy pro konkrétní uložiště.

Každé publikované uložiště musí obsahovat manifest obsahující seznam názvů a „hash“ hodnot pro všechny certifikáty a podepsané objekty (CER, CRL, ROA), které jsou v danou chvíli publikovány danou CA.

Certifikáty a ROA, které jsou z nějakého důvodu aktualizovány, mohou nést shodný název se starší verzí tohoto certifikátu. V tomto případě původní objekt bude přepsán.

Repositář je dále spojen s RPKI validátorem za pomocí nástroje rsync (kapitola 4.3), pomocí kterého si validátor (4.4) stáhne z repositáře jednotlivé certifikáty a ověřené objekty (Obrázek **5**). Následně si pomocí RPKI/rtr protokolu (kapitola 3.1.8) předá validátor se směrovačem informace o validaci jednotlivých síťových prefixů.



Obrázek 5: Struktura přeposílání informací od repositáře po směrovač

3.1.8 RPKI/Router protokol

Standardem pro tento protokol je dokument RFC6810 [17]. Směrovač s podporou pro RPKI navazuje spojení s lokální mezipamětí validátoru, která obsahuje všechny ověřené objekty stažené z daných repositářů. Vztah mezi směrovačem a mezipamětí validátoru je typu Client/Server, přičemž je udržováno otevřené a (ověřené) spojení. Výměna informací (posloupnosti PDU – Protocol Data Unit) mezi nimi probíhá pomocí RPKI/RTR.

Jednotlivá PDU mohou obsahovat tyto pole:

Protocol version*:	verze protokolu
PDU Type*:	označuje typ PDU
Serial number:	aktuální verze obsahu mezipaměti. Tato hodnota se mění pokaždé, když mezipaměť aktualizuje svá data z RPKI repositáře
Session ID:	číslo spojení mezi mezipaměťí a konkrétním směrovačem
Length*:	tato hodnota určuje počet bajtů celého PDU včetně hlavičky
Flags:	hodnota posledního bitu v tomto poli je 1 pro oznámení a 0 pro stažení již dříve oznámeného IPvX Prefix PDU s úplně stejným prefixem, délkou, maximální délkou a číslem ASN
Prefix Length:	definuje nejkratší povolený prefix pro danou IP adresu
Max Length:	definuje maximální délku prefixu pro danou IP adresu. Nesmí být však menší než Prefix Length.
Prefix:	IPv4 nebo IPv6 adresa pro daný objekt ROA
ASN	číslo autonomního systému, který smí užívat daný prefix.
Zero:	rezervovaná hodnota pro případné změny v protokolu. Tato hodnota musí být nulová a je při převzetí ignorována.

pole označené * musí být vždy v PDU obsaženy

Typy PDU:

Serial Notify (typ PDU = 0):	oznámení, že mezipaměť obsahuje nová data
Serial Query (typ PDU = 1):	žádost směrovače o všechny PDU, které mají vyšší hodnotu sériového čísla než je sériové číslo v „Serial query“
Reset Query (typ PDU = 2):	žádost směrovače o přeposlání celé aktuální databáze z mezipaměti
Cache Response (typ PDU = 3):	odpověď na „Serial Query“ nebo „Reset Query“ počtem aktuálních PDU
IPv4 Prefix (typ PDU = 4):	samotné přeposlání IPv4 prefixu
IPv6 prefix (typ PDU = 6):	samotné přeposlání IPv6 prefixu
End of Data (typ PDU = 7):	oznámení od mezipaměťi, že již neobsahuje další nová data

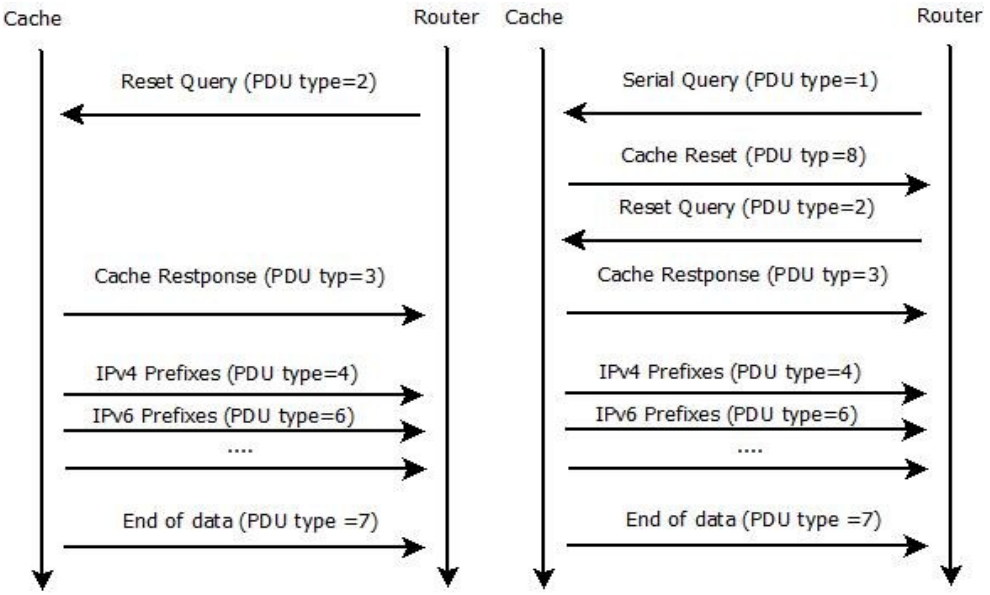
Cache Reset(typ PDU = 8):	odpověď od mezipaměťi o nemožnosti poskytnout inkrementální aktualizaci, která by začínala stejnou hodnotou „Serial Number“ určenou směrovačem.
Error Report (typ PDU = 9):	nahlášení chyby na kterékoliv straně

K výměně PDU mezi směrovačem a mezipaměťí dochází ve dvou případech.

- Při spuštění nebo restartu směrovače – (Obrázek **6**)
- Při jakékoli změně (aktualizaci) v mezipaměti - (Obrázek **7**)

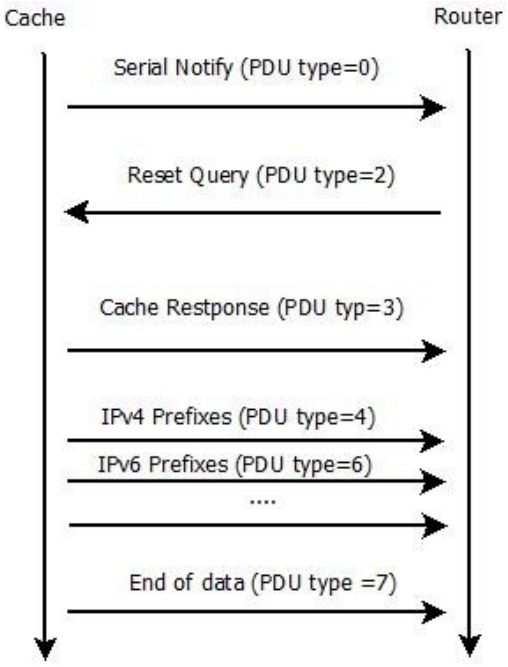
V prvním případě, kdy směrovač navazuje spojení při jeho startu, nebo když selže dřívější spojení s mezipamětí, směrovač odešle zprávu „Reset Query“ s hodnotou „Serial Number“, která je nulová. Mezipaměť odpovídá zprávou „Cache response“, kterou potvrzuje požadavek směrovače a následně začne odesílat samotná data pomocí PDU typu 4 nebo 6. V tomto PDU je také obsažena hodnotu flag, jejíž poslední bit říká směrovači, zda si má tuto adresu přidat či odebrat z BGP tabulky. Po odeslání všech dat informuje mezipaměť směrovač pomocí zprávy „End of Data“, čímž se změní hodnota „Serial Number“. V případě, kdy selže již navázané spojení mezi těmito stranami, směrovač odešle zprávu „Serial Query“, která obsahuje pole „Session ID“ předešlého spojení, aby se zajistilo stejných hodnot „Serial Number“, které byly nastaveny při poslední aktualizaci. K omezení doby, po kterou musí mezipaměť držet data potřebná ke generování inkrementální aktualizace, musí směrovač zaslat zprávu „Serial Query“ nebo „Reset Query“ alespoň jednou za hodinu. Z toho plyne, že mezipaměť většinou neudržuje tato data více než jednu hodinu.

V případě, že směrovač obdrží po odeslání zprávy „Serial Query“ zprávu „Cache Reset“, znamená to, že mezipaměť nemůže poskytnou inkrementální aktualizaci pro „Serial Number“ směrovače. Tato situace může nastat například z důvodu, že směrovač neposlal „Serial Query“ nebo „Reset Query“ včas a mezipaměť vyčistila stará nepotřebná data. V tomto případě by se měl směrovač pokusit o připojení k další mazipaměťi, kterou má nastavenu ve svém seznamu mezipamětí. Pokud však není žádná k dispozici, musí směrovač zaslat znovu zprávu „Reset Query“, čímž získá všechna data z dané mezipaměti.



Obrázek 6: Komunikace mezi směrovačem a RPKI validátorem při startu směrovače (vpravo) nebo při rozdílné hodnotě „Serial Number“ (vlevo)

V druhém případě, kdy dojde ke změně v mezipaměti, oznámí tato mezipaměť pomocí zprávy „Serial Notify“ směrovači, že obsahuje nová data a očekává od směrovače zprávu „Serial Query“. Dále se mezipaměť chová stejným způsobem jako v předešlých případech.

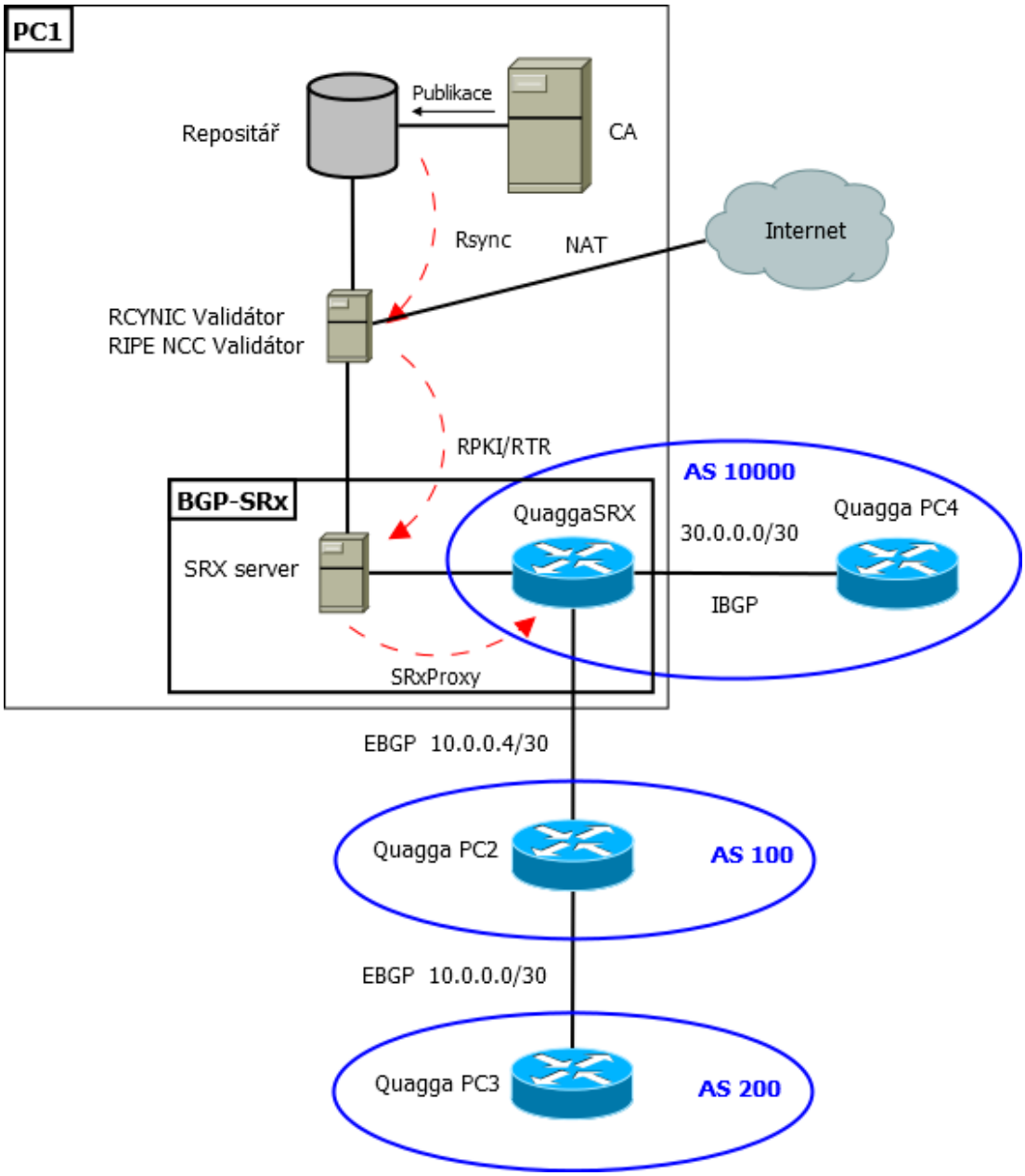


Obrázek 7: Komunikace mezi směrovačem a mezipaměti validátoru při změně v mezipaměti.

4 Praktická část

Practická část této práce spočívá v nasazení CA a jednotlivých prvků RPKI, kterými jsou RPKI validátory a směrovače, do testovacího provozu. Topologii nasazení jednotlivých prvku můžeme vidět na Obrázek **8**.

4.1 Topologie testovacího prostředí



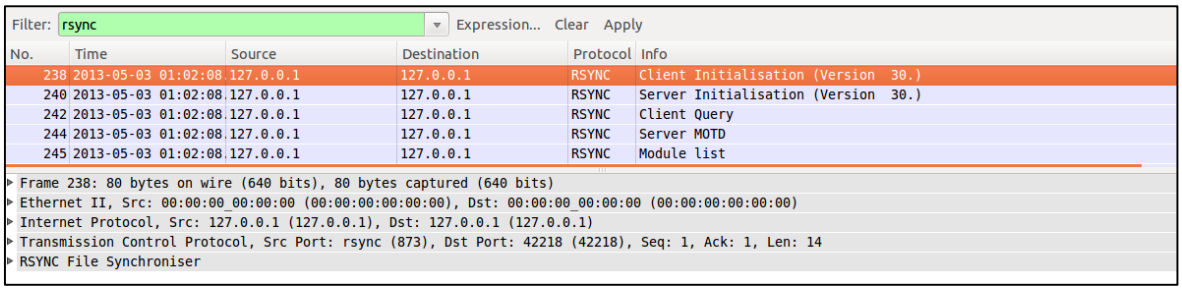
Obrázek 8: Topologie znázorňující propojení jednotlivých VMware stanic PC1, QuaggaPC2, QuaggaPC3 a QuaggaPC4

Pro testování a provoz výše zmiňovaných prostředků k ověření autorizace propagace síťových prefixů bylo použito čtyř Vmware strojů PC1, Quagga PC2, Quagga PC3, Quagga PC4 na kterých byl nainstalován operační systém Ubuntu 12.04 a které jsou navzájem propojeny přes virtuální síťová rozhraní. Jednotlivé Vmware stroje jsou k dispozici na přiložených DVD. Celá naše topologie (Obrázek **8**) byla testována na jediném PC, které bylo této konfigurace: CPU IntelCore2 Duo E8400 3GHz, paměť 8GB RAM, Operační systém: Windows7 64-bit. 2xHDD v RAID 0 z důvodu rychlosti přístupu na disk. Přihlašovací jméno a heslo do všech Vmware strojů je **„quagga“**

Obsah Vmware PC1:

- Certifikační autorita spolu s rcynic validátorem a jednotlivými nástroji
- NCC RIPE validátor
- SRx server
- Implementace směrovače QuaggaSRx, která je nastavena pro autotomní systém 10000

Na tomto virtuálním stroji je nainstalována CA, která generuje jednotlivé certifikáty pomocí OpenSSL a ukládá je do svého repositáře. Validátory RIPE NCC validátor (běží na portu: 8282) a RCYNIC validátor si můžou z tohoto repositáře pomocí protokolu Rsync stáhnout jednotlié certifikáty do svoji mezipaměti a tyto certifikáty ověřit. (zachycenou komunikaci, pomocí wireshark, mezi repositáři a validátorem můžeme vidět na Obrázek **9**) K validátoru se pomocí RPKI/RTR protokolu připojí v našem případě SRx server který, běží na portu 17900. SRx server si zpracuje jednotlivá data přijatá od validátoru. K tomuto serveru je připojený směrovač QuaggaSRx AS10000, který s tímto serverem komunikuje pomocí protokolu SRxProxy. QuaggaSRx přeposílá tomuto serveru požadavky na ověření jednotlivých příchozích zpráv UPDATE, které obdržel od svých sousedů a na základě nich SRx server tyto požadavky ověří a výsledky ověření pošle směrovači.



Obrázek 9: Zachycení komunikace mezi repositářem a RPKI validátorem, která probíhá pomocí „Rsync“

Abychom mohli tyto zprávy obdržet je potřeba nastavit do konfigurace jednoho ze směrovačů jednotlivé sítě, které se pomocí nastavení statické cesty, příkazem „ip static route 0.0.0.0/0 0.0.0.0“, rozešlou na ostatní směrovače. Nastavení jednotlivých sítí a statické cesty bylo nastaveno na virtuální stanici Quagga PC3.

Ukázka konfigurace směrovače QuaggaSRx:

```
!
hostname ASN10000-SRX
password zebra
log stdout
!
router bgp 10000
 bgp router-id 10.0.0.5
 neighbor 10.0.0.6 remote-as 100
 neighbor 30.0.0.2 remote-as 10000

 ! SRx Basic Configuration Settings
 srx set-proxy-id 10.0.0.5                  - identifikace smněrovače
 srx set-server localhost 17900             - nastavení adresy SRx serveru
 srx keep-window 900                        - udržení informace o validaci [s]
 srx evaluation origin_only                 - povolení zpracování politik
 srx display                                - zapnutí zobrazení SRX informací

 ! SRx Evaluation Configuration Settings
 srx set-origin-value undefined     - nastavení výchozí hodnoty pro ověření původu
 srx policy prefer-valid            - upřednostnění validních prefixů od ostatních
 srx policy ignore-notfound         - ignorování prefixů s notfoud stavem

 ! Connect to SRx-server
 srx connect                        - připojení k SRx serveru
!
line vty
!
```

Virtuální stanice QuaggaPC2 pracuje s implementací směrovače Quagga s nastaveným číslem autonomního systému 100, který je následující konfigurace.

```
hostname AS100
password zebra
log stdout
!
router bgp 100
 bgp router-id 10.0.0.6
 network 10.0.0.0/30
 network 10.0.0.4/30
 neighbor 10.0.0.2 remote-as 200
 neighbor 10.0.0.5 remote-as 10000
!
line vty
```

Quagga PC3 slouží jako generátor jednotlivých síťí a leží v AS 200. Těmito sítěmi jsou 20.0.0.0/24 až 20.255.255.0/24 což nám dává dohromady 65536 síťí. Tento směrovač je sousedem směrovače Quagga, který leží v AS100

Konfigurace:

```
hostname AS200
password zebra
log file /var/log/quagga/bgpd.log
log stdout
!
debug bgp events
debug bgp updates
!
router bgp 200
 bgp router-id 10.0.0.2
 network 10.0.0.0/30
 network 20.0.0.0/24
 network 20.0.1.0/24
 network 20.0.2.0/24
 network 20.0.3.0/24
 ---
 network 20.255.255.0/24
neighbor 10.0.0.1 remote-as 100
!
line vty
!
```

Na virtuálním stroji Quagga PC4 je nainstalován směrovač Quagga, který je BGP sousedem od QuaggaSRx směrovače. Oba tyto směrovače jsou ve stéjném AS10000. Tento směrovač slouží k ověření, že skutečně zafungovali jednotlivé politiky aplikované na směrovači QuaggaSRX.

Konfigurace:

```
hostname AS10000(30.0.0.2)
password zebra
log stdout
!
router bgp 10000
 bgp router-id 30.0.0.2
 neighbor 30.0.0.1 remote-as 10000
!
line vty
!
```

4.2 Nasazení certifikační autority

Dostupnou implementací nástroje pro samotnou zprávu a pro RPKI infrastrukturu v laboratorních podmínkách je generování jednotlivých certifikátů certifikační autorita (CA), která se nachází na adrese projektu [18], kde je také podrobná dokumentace. Implementace tohoto nástroje začala vznikat v roce 2006. Do roku 2008 byla vyvíjena za pomoci finančních prostředků ARIN ve spolupráci s jinými RIR. V současné době je vývoj implementace financován ministerstvem vnitřní bezpečnosti U.S. (DHS). Součástí této implementace je také řada nástrojů. Pro nás jsou stěžejní tyto nástroje.

- rcynic – nástroj pro ověřování objektů, který ověřuje jednotlivé podpisy, expirační doby a shodu s profily RPKI. Po samotné validaci se vytvoří XML soubor rcynic.xml, který obsahuje výstup jednotlivých výsledků validace z validátoru.

- rcynic-html - nástroj pro převod XML stavového výstupu na HTML stránky zobrazující stav a historii validátoru.

- rtr-origin – implementace rpki-RTR protokolu, jejímž vstupem je výstup validátoru „rcynic“. Součástí tohoto nástroje je rpki-RTR server a testovací klient pro zkoumání obsahu databáze vytvořené z údajů poskytnutých validátorem.

- rcynic-cron – skript, který pomocí plánovaného spouštění úloh umožní spustit zmíněnou sadu nástrojů.

Nástroj CA můžeme nainstalovat dvěma způsoby. Instalaci můžeme provést ručně, nebo za pomocí nástroje apt-get. V případě, že si vybereme první variantu, musíme doinstalovat všechny potřebné balíky pro samotnou kompilaci. Všechny tyto potřebné balíky můžeme najít na výše uvedené adrese. Tento krok, ale vyžaduje trpělivost, protože některé balíky musí být instalovány v dané konkrétní verzi.

Pro instalaci pomocí nástroje apt-get musíme nejdříve nakonfigurovat cestu pro balíky „rpki-ca“ samotnou CA a „rpki-rp“, který obsahuje jednotlivé nástroje.

Nejprve si nainstalujeme veřejný klíč pro daný repositář obsahující tyto balíky.

```
        wget -q -O - http://download.rpki.net/APT/apt-gpg-key.asc |
sudo apt-key add -
```

Dále si vytvoříme soubor obsahující cesty k jednotlivých balíkům

```
$ sudo wget -q -O /etc/apt/sources.list.d/rpki.list
http://download.rpki.net/APT/rpki.ubuntu.list
```

Aktualizujeme seznam dostupných balíků:

```
$ sudo apt-get update
```

Provedeme samotnou instalaci:

```
$ sudo apt-get install rpki-rp rpki-ca
```

Při této instalaci se nám také nainstalují binární balíky, které jsou vyžadovány pro správný běh jednotlivých nástrojů a CA. Výjimkou je pouze balík OpenSSL, který se nenainstaluje kompletně, chybí mu podpora X.509 pro čísla autonomních systémů a IP adres. Proto musíme konfiguraci a instalaci OpenSSL provést ručně a zprovoznit tak dané rozšíření. Toto provedeme pomocí následujících příkazů v privilegovaném režimu:

```
./config enable-rfc3779
make depend
make
make test
make install
```

Pro tuto diplomovou práci je použito balíku OpenSSL ve verzi 1.0.0i, který je volně dostupný na adrese http://www.openssl.org, a je zároveň plně kompatibilní pro danou implementaci RPKI CA.

Před samotným spuštěním nástroje, musíme CA přiřadit čísla autonomních systémů a IP adres, aby je měla CA pod svou správou a mohla je dále rozdělovat dalším subjektům. Abychom tohoto docílili, musíme nejdříve vytvořit „Self-Signed“ certifikát s rozšířením pro IP adresy a AS a tímto alokovat dané CA tyto zdroje. Tohoto docílíme vytvořením certifikátu pomocí OpenSSL. Před samotným vytvořením tohoto certifikátu musíme vytvořit konfigurační soubor *.conf.

```
[req]
default_bits            = 2048      - výchozí velikost klíče v bitech
default_md              = sha256    - použitý algoritmus pro šifrování
distinguished_name      = req_dn    - vodkaz na sekci s žádosti o certifikát
prompt                  = no        - zakázání dotazování certifikačních hodnot,
                                      nabývá hodnot z [req_dn]
```

```
[req_dn]
CN                      = ROOT       - commonName

[root_x509v3_extensions]
basicConstraints        = critical,CA:true
subjectKeyIdentifier    = hash
keyUsage                = critical,keyCertSign,cRLSign
subjectInfoAccess       = @sia
certificatePolicies     = critical,1.3.6.1.5.5.7.14.2
sbgp-autonomousSysNum   = critical,@rfc3779_asns
sbgp-ipAddrBlock        = critical,@rfc3997_addrs

[sia]             - pole určující cestu k adresáři vystaveného certifikátu
1.3.6.1.5.5.7.48.5;URI  = rsync://localhost/rpki/
1.3.6.1.5.5.7.48.10;URI = rsync://localhost/rpki/root.mft

[rfc3779_asns]    - pole určující přidělená čísla AS
AS = 0-4294967295
[rfc3997_addrs]   - pole určující rozsah přidělnených adres
IPv4 = 0.0.0.0/0
IPv6 = ::/0
```

Hodnoty obsažené v `[root_x509v3_extensions]`:

```
basicConstraints        - nastavení certifikátu na certifikát CA
subjectKeyIdentifier    - sleduje obecné zásady RFC3280
keyUsage                - použití certifikátu
subjectInfoAccess       - odkaz na sekci [sia]
certificatePolicies     - certifikační politika
sbgp-autonomousSysNum   - odkaz na [rfc3779_asns]
sbgp-ipAddrBlock        - odkaz na [rfc3997_addrs]
```

Následovně je zapotřebí si vygenerovat vlastní privátní klíč, který bude použit k podepsání našeho certifikátu.

```
$ openssl genrsa -out root.key 2048
```

Vygenerovat žádost o certifikát.

```
$ openssl req -new -config root.conf -out root.req -key
root.key
```

Na základě žádosti a námi vygenerovaného privátního klíče vytvoříme vlastní certifikát:

```
$ openssl x509 -req -sha256 -signkey root.key -in root.req -
outform DER -out root.cer –extfile root.conf –extensions
x509v3_extensions -days 1825
```

Pro ověření správného přidělení IP adres a čísel AS a tím ověření rozšíření RFC3779 požijeme následující příkaz.

```
$ openssl x509 -text -inform DER -in root.cer
```

Pro sprvávný běh používá CA tyto démony:

RPKID

Reprezentuje hlavní démon RPKI. Konfigurace RPKID je proces o dvou krocích: nejprve úprava config souboru, aby byl RPKID zaveden do bodu, kde bude schopen komunikovat pomoci left-right protokolu a potom dynamická konfigurace démona pomocí tohoto protokolu.
Druhý krok je řešen pomocí nástroje příkazové řádky „rpkic“ nebo přes webové rozhraní. RPKID si ukládá dynamická data v SQL databázi, která pro něj musí být vytvořena. Toto je popsáno v návodu pro nastavení MySQL.

PUBD

PUBD je publikační démon. Implementuje serverovou část publikačního protokolu a je používán Rpkid démonem na publikaci certifikátů a jiných objektů, které Rpkid generuje. Tento démon ukládá data v SQL databázi, která pro něj musí být vytvořena. PUBD také ukládá publikované certifikáty a kryptografické objekty (.cer, .roa, .mft, .crl) do adresáře, který je nastaven v rsync.conf.

ROOTD

Jedná se o implementaci serverové části „up-down“ protokolu. Je to separátní program protože kořenový certifikát RPKI certifikátu vyžaduje speciální zacházení a rovněž může požadovat speciální manipulační politiku. ROOTD je jednoduchá implementace určená k testovacímu použití, není vhodný pro použití v produkčním prostředí.

Rpkic

Jedná se o command line interface, pomocí kterého můžeme nastavovat CA. K tomu nám slouží i grafiké prostředí, které je dostupné v případě naší lokální CA na URL: https://127.0.1.1/accounts/login/

Nastavení pro CA je obsaženo v jednom sdíleném souboru “/etc/rpki.conf“, ve kterém je možné nastavit chování jednotlivých démonů.

Již dříve vygenerovaný certifikát root.cer musí být zkopírován do složky /usr/share/rpki/publication.root a privátní klíč, kterým je tento certifikát podepsán musí být zkopírován do složky /usr /share/rpki/, jak je definováno v konfiguračním souboru.
Po instalaci můžeme certifikační autoritu spustit pomocí příkazu:

```
$ sudo service rpki-ca start
```

Pomocí CA jsem si alokoval pro certifikační autoritu „root“ rozsahy ip adres a čísla AS, které jsem dále přiřadil mezi jednotlivé potomky. Abychom mohli akolovat pro naší CA požadované zdroje je potřeba pomocí nástroje „rpkic“ použít tyto příkazy:

- vytvoříme a nastavíme (handle).repository-request.xml soubor

```
$ sudo rpkic initialize
```

- dále vytvoříme a nastavíme (handle).repository-response.xml

```
$ sudo rpkic configure_publication_client (handle).repository-
request.xml

$ sudo rpkic configure_repository (handle).repository-
response.xml
```

Na místo hodnoty (handle) je použito „handle“ hodnota specifikováná v rpki.conf . V našem případě je tato hodnota nastavena jako **„root“**.

Abychom mohli otestovat funkčnost a zatížení topologie musíme pro alokované zdroje jednotlivých subjektu pod CA vytvořit jednotlivá ROA. Abychom nemuseli pro každou síť vytvářet ROA postupně, byl vytvořen soubor roas.csv pomocí kterého v GUI CA vytvoříme jednotlivé ROA. GUI CA můžeme najít na adrese https://127.0.1.1/rpki/, kde nastavené přihlašovací jméno a heslo je **„root“** Soubor roas.csv se nachází na přiložením DVD. Jelikož jsme si vybrali celkový počet síťí 65536 a pro takový počet nebylo možné vytvořit jednotlivé ROA z důvodu velkého zatížení procesoru a nestability systému, tak se tento soubor zredukoval pouze na 11189 ROA.

4.3 Rsync

Rsync je nástroj pro unixové systémy. S jeho pomocí lze synchronizovat soubory a adresáře z jednoho umístění do jiného, přičemž minimalizuje přenosy pomocí delta rozdílů množin a pomocí kontrolního součtu umí porovnat skutečný obsah těchto uložišť. Výměna dat probíhá prostřednictvím démonu RSYNC, který naslouchá pomocí TCP protokolu na portu 873. Tento nástroj využívá RPKI validátor k tomu, aby si do své lokální mezipaměti stáhl certifikáty z veřejných repositářů od jednotlivých certifikačních autorit.

Abychom se mohli připojit k našemu repositáři, je potřeba nastavit konfigurační soubor pro rsync démona. Konfigurační soubor se nachází v adresáři /etc/rsyncd.conf a start démonu provedeme příkazem:

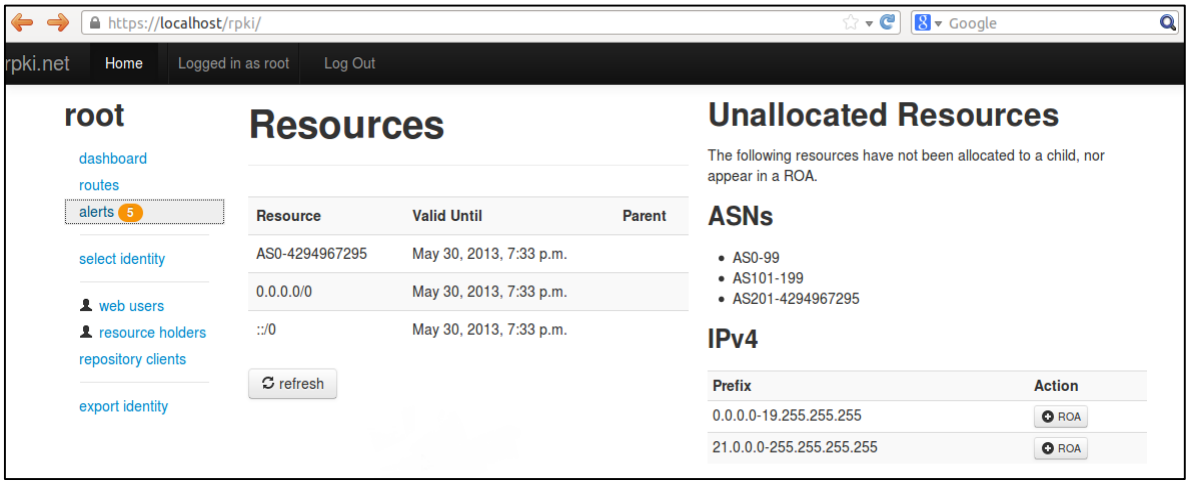
```
$ sudo rsync --daemon
```

Ukázka konfiguračního souboru:

```
log file    = /var/log/rsyncd
pid file    = /var/run/rsyncd.pid
uid         = nobody
gid         = nogroup
address     = localhost
[rpki]
      use chroot         = no
      read only          = yes
      transfer logging   = yes
      path               = /usr/share/rpki/publication
      comment            = RPKI ROOT

[root]
      use chroot         = no
      read only          = yes
      transfer logging   = yes
      path               = /usr/share/rpki/publication.root
      comment            = RPKI ROOT
```

Obrázek níže nám znázorňuje grafické prostředí pro CA, kde se nachází alokované rozsahy IP adres a čísel AS, které má daná CA přidělena, které pak může dále rozdělovat svým podřízeným subjektům. Dále pak v pravém dolním rohu obrázku vidíme, že pro jednotlivé rozsahy, které nejsou stále přiděleny jiným subjektům, můžeme vytvořit další ROA.



Obrázek 10: Certifikační autorita s alokovanými čísly AS a IP rozsahy adres

4.4 RPKI Validátory

Validátor slouží k načtení certifikátů, seznamů odvolaných certifikátů, manifestu a ROA z publikovaných repositářů od jednotlivých certifikačních autorit do lokální Cache daného validátoru. Následně validátor ověří tyto stažené soubory a distribuuje výsledky ověření. Tyto výsledky jsou následně použity pro rozhodování směrovače při směrování BGP informace. Stahování obsahu repositářů probíhá pomocí nástroje „rsync“, na základě Trust Anchor Locator, který obsahuje rsync URI a veřejný klíč dané certifikační autority. V dnešní době jsou dostupné tyto validátory:

- RCYNIC validátor [18]
- RIPE NCC RPKI validator [19]
- Raytheon BBN: RPSTIR Project [20] - jedná se o validační nástroj od společnosti Raytheon BBN.

Pro tuto diplomovou práci byly vybrány „RIPE NCC RPKI“ validátor ve verzi 2.8.1 a „RCYNIC“ validátor. Tyto validátory byly vybrány z důvodu jejich rozšíření v dnešním světě ke kterým je v dnešní době k dispozici i testovací implementace.

4.4.1 RIPE NCC RPKI Validátor

Validátor RIPE NCC běží pod Unixovými systémy s podporou JAVA a rsync. Pro dobrou přehlednost tento validátor obsahuje také GUI (Obrázek **11**). Před tím, než samotný validátor spustíme, musíme validátoru definovat TAL soubory pro jednotlivé certifikační autority.

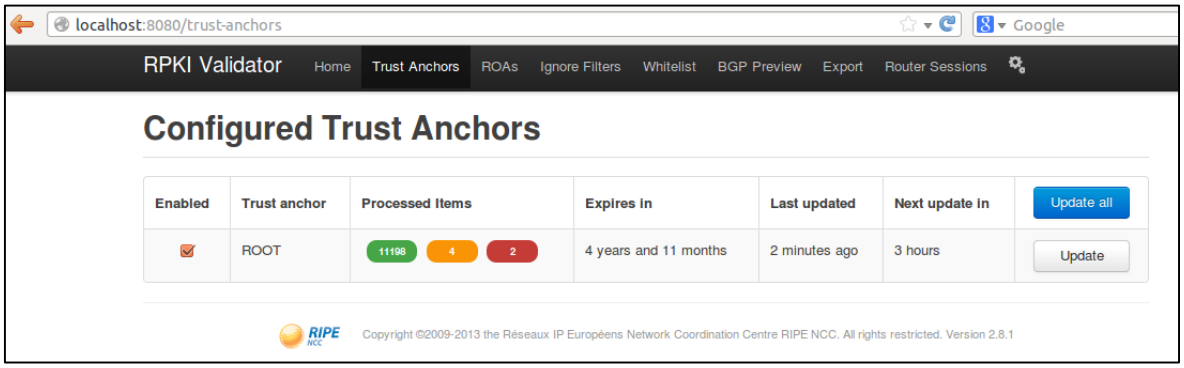
Abychom se mohli připojit k repositářům jednotlivých certifikačních autorit, musíme k jednotlivým repositářům vytvořit soubor *.tal podle konvence RIPE NCC RPKI validátoru, který pak musí být obsažen v adresáři /conf/tal/ . Tento TAL obsahuje povinné hodnoty „ca.name“, které definujeme jméno, pod kterým vystupuje daný TAL v rámci GUI validátoru. Dále obsahuje hodnoty „certificate.location“ a „publickey.information“, které představují rsync URI a veřejný klíč k certifikátu v daném repositáři. Hodnota „prefetch.uris“ je zde nepovinou hodnotou, kterou můžeme specifikovat, že chceme pro validaci použít jen některé adresáře publikované v daném repositáři. Tyto specifické rsync URI musí být v této hodnotě odděleny čárkou. Jednotlivé TAL soubory, které odkazují na repositáře jednotlivých RIR můžeme najít na přiloženém DVD3.

Příklad TAL odkazující na lokální repositář naší CA

```
ca.name = ROOT
certificate.location = rsync://localhost/root/root.cer
public.key.info=
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA36HLJwWY+LKgtZtttf6mJNHp
nqnBRHLH//8zyMJJ5XhRbww0FEbnpvz7ox7y09EEUxKjN7p4QHxzpx27AVDK67mcZJuI
iXs6T2gS7eXi56nK2X2Ma6hhAFpazVmOE/p+ueI7oxoRHJ8fvVLtnnUIKoCQ2hLzvShl
353Oyq/ZtaOmM+aWIWI1aATSTYYbfZ+hOuu7SGp+q45+HX7g2t6w2DqmDmInLr5cwg7w
sTczUbZDphtQItwH92yqZiXqaG/z2yjCOO82kIi4WtmwNCg0jdY0OvDsUO/Ix/9WqcKD
iCU/wV+8tKc77gqJgUGVo6EzbyhOv9OsHQXA+NbzZ8Kv5wIDAQAB
```

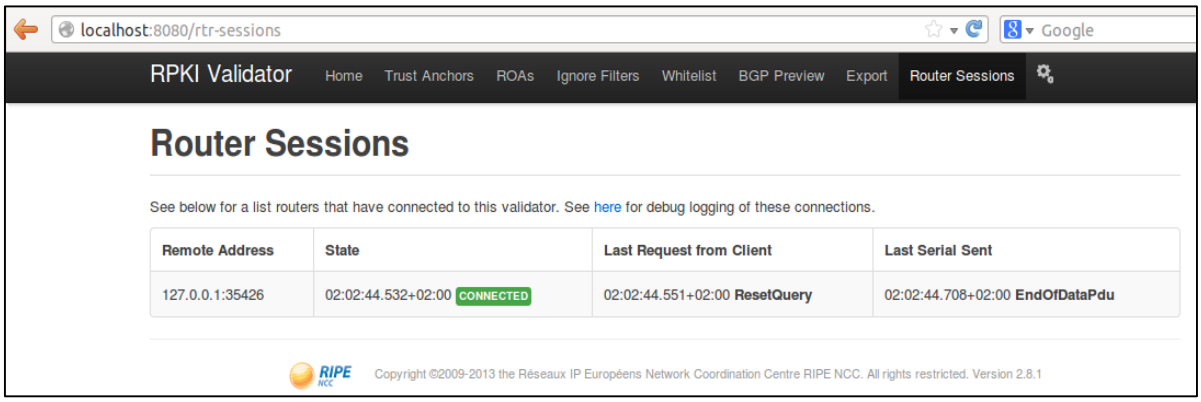
Pro start validátoru nám slouží soubor rpki-validátor, který se nachází v adresáři /bin. Samotné spuštění validátoru, musíme provést z jiného adresáře, než je samotný spustitelný soubor. (`$ ./bin/rpki-validator`)

Na následujícím obrázku vidíme úspěšné připojení validátoru k našemu repositáři, kde vidíme počet validních cest, platnost certifikátu podle, kterého jsou jednotlivé cesty validovány a dále pak informace o aktualizaci, kdy naposledy proběhla a kdy proběhne další.



Obrázek 11: Ukázka RPKI validátoru s načtenými objekty z repositáře

Abychom mohli tyto validní cesty dále propagovat do směrovačů je zapotřebí připojit k tomuto validátoru buďto směrovač s podporou RPKI nebo SRX server, který je zdokumentovaný níže. Následující Obrázek **12** nám znázorňuje úspěšné navázání spojení validátoru se SRX serverem. Můžeme zde vidět vzdálenou adresu SRX serveru, kdy se tento server připojil, kdy byl validátor naposledy dotazován a také kdy naposledy propagoval informace.



Obrázek 12: Navázané spojení SRxServeru s RIPE RPKI validátorem

4.4.2 Rcynic Validátor

Jak již bylo zmíněno, validátor Rcynic je dostupný na adrese trac.rtpki.net. Tento validátor je závislý na OpenSSL kryptografické knihovně „libcrypto“, která implementuje šifrovací algoritmy. Validátor Rcynic můžeme spustit v adresáři, ve kterém se nachází konfigurační soubor rcynic.conf, nebo musíme specifikovat cestu k souboru pomocí volby **–c**. V tomto konfiguračním souboru specifikujeme, jak se má RCYNIC chovat. Konfigurační proměnné v samotném konfiguračním souboru jsou uvedeny na adrese https://trac.rpki.net/wiki/doc/RPKI/RP/rcynic. V našem konfiguračním souboru byly použity tyto proměnné:

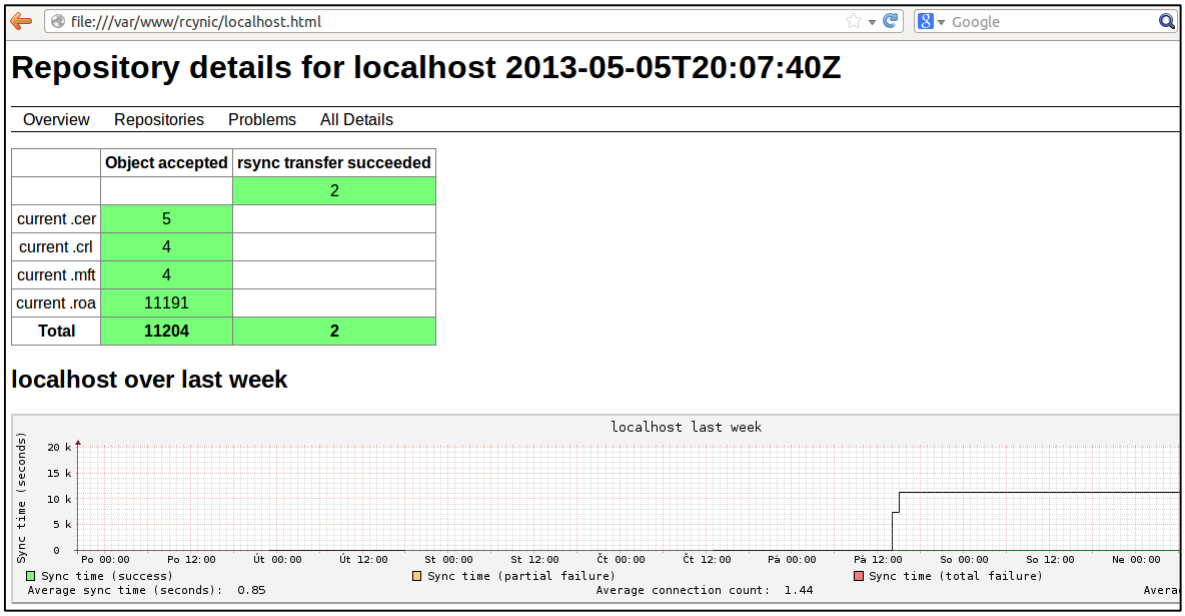
- **rsync** - program – nastavení cesty k programu rsync
- **authenticated** – cesta k adresáři, kde má rcynic odkládat ověřené objekty.
- **unauthenticated** – cesta k adresáři, kde rcynic odkládá neověřená data. Do tohoto adresáře jsou ukládána všechna data z jednotlivých repositářů, aby se při příštím spuštění omezilo síťový provoz mezi jednotlivými repositáři a validátorem.
- **xml-summary** – nastavení cesty pro soubor formátu XML, kde se na konci fáze průběhu validace uloží stavový výstup rcynic.
- **jitter** – hodnota udávající interval v sekundách, z tohoto intervalu si rcynic zvolí náhodné číslo a tuto dobu čeká. Po této době dochází k samotné validaci. jitter je využit z důvodu rozdělení zátěže, aby se rsync servery nepřetěžovaly v případě velkého počtu klientů. Výchozí hodnotou je 600.
- **log-level** – nastavení logování. Jednotlivými logy mohou být nastaveny na tyto hodnoty:
  - `log_sys_err` - chyba z operačního systému nebo knihovny
  - `log_usage_err` - špatné použití (lokální chyba konfigurace)
  - `log_data_err` - špatná data (poškozený certifikát nebo CRL)
  - `log_telemetry` - normální výpis o postupu rcynic
  - `log_debug` - užitečné pouze při ladění
- **trust-anchor-locator** – nastavení cest k souborům *.tal, které obsahují rsync URL

Příklad:

```
[rcynic]
rsync-program       = /usr/bin/rsync
authenticated       = /var/rcynic/data/authenticated
unauthenticated     = /var/rcynic/data/unauthenticated
xml-summary         = /var/rcynic/data/rcynic.xml
```

```
jitter                      = 600
log-level                   = log_debug
trust-anchor-locator.1      = /etc/rpki/trust-anchors/root.tal
trust-anchor-locator.2      = /etc/rpki/trust-anchors/afrinic.tal
```

Validátor RCYNIC poskytuje také grafický výstup pro ověřované objekty a certifikáty. Samotný grafický výstup je znázorněn na Obrázek **13**. Na obrázku můžeme vidět i graf s vývojem ověřovaných objektů za poslední týden.



Obrázek 13: Ukázka výpisu ověřených objektů validátorem Rcynic

4.5 Implementace RPKI architektury na směrovacích prvcích

RPKI je oficiálně podporován na platformách CISCO, JUNIPER a implementace Quagga směrovače s Secure Routing Extension (SRX).

4.5.1 Cisco

Směrovače Cisco mají implementovanou RPKI podporu pro řady směrovačů ASR1000, ASR903, ASR 901, 7200, 7600 s IOS ve verzích 15.2 (1)S, 15.2 (4)S.

RIPE poskytuje ukázkový směrovač CISCO, který je k dispozici na „rpki-rtr.ripe.net“ připojením pomocí „TELNET“, kde přihlašovací jméno je „ripe“ a heslo je prázdné.

4.5.2 Juniper

„Juniper“ směrovače oficiálně podporují RPKI od verze JunOS 12.2. U této platformy RIPE opět poskytuje veřejné testovací směrovače, které jsou také k dispozici připojením pomocí „TELNET“ na adrese 193.34.50.25 a 193.34.50.26, kde přihlašovací jméno je „rpki“ a heslo „testbed“.

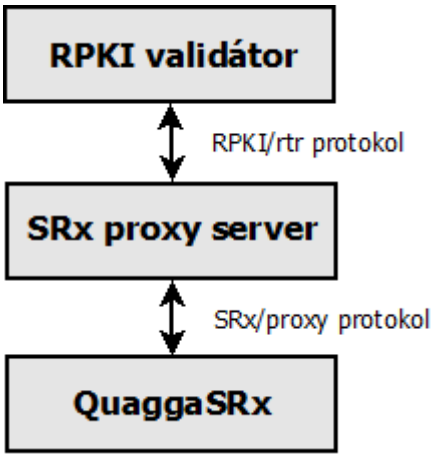
Vzhledem k tomu, že výše uvedené směrovače Cisco ani Juniper naše laboratoř neobsahuje, jedinou dostupnou implementací, kterou lze nasimulovat a otestovat prostředí RPKI byla implementace softwarového směrovače Quagga, pro kterou existuje rozšiřující modul BGP Secure Routing Extension (BGP-SRx).

4.6 BGP-SRx

BGP-SRX je open source implementace, která v dnešní době se stále vyvíjí v národním institutu pro standarty a technologie NIST spolupracující s ministerstvem pro vnitřní bezpečnost DHS. Architektura BGP-SRX obsahuje tři části. SRx Server, SRx API a Quagga SRx, ve které je integrovaný SRxAPI. Jednotlivé balíky pro tuto architektu se nachází na stránkách tohoto projektu [21]. U těchto balíků můžeme najít také podrobné manuály, jak tyto implementace uvést do chodu.

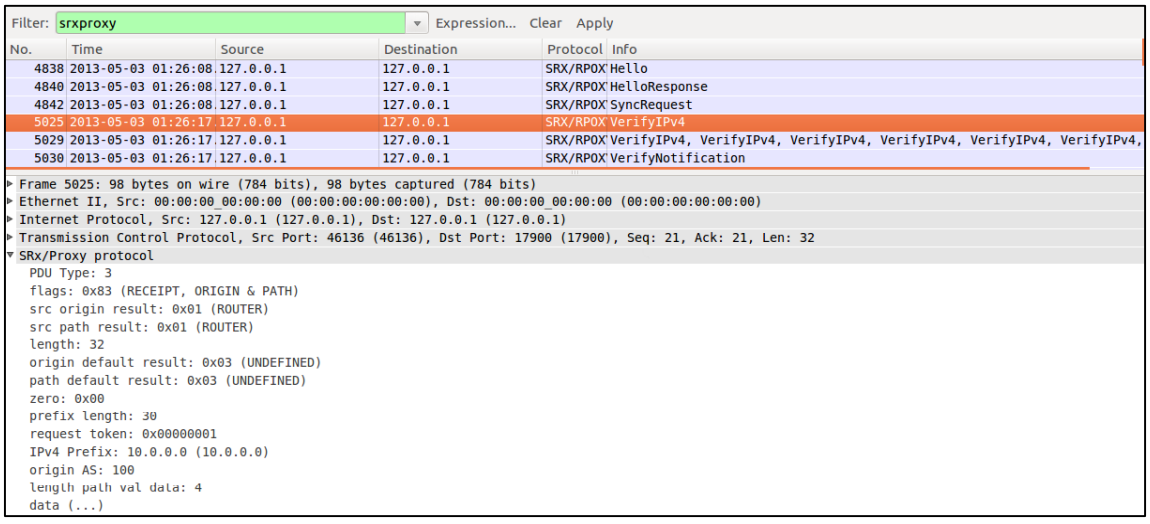
4.6.1 SRx server

Jedná se o Proxy server, zprostředkovávající komunikaci mezi Quaggou SRx a RPKI validátorem. Tato komunikace je znázorněna na Obrázek **14**, ze kterého je patrné, že přenos informací mezi RPKI validátorem a SRx serverem je zajištěn pomocí protokolu RPKI/rtr. SRX server pak dále komunikuje s Quaggou pomocí SRXproxy protokolu [22], který je založen na TCP protokolu. Podrobný manulál můžeme najít opět na stránkách projektu [23]. SRx server spustíme z adresáže /usr/bin/ příkazem `$srx_server`.

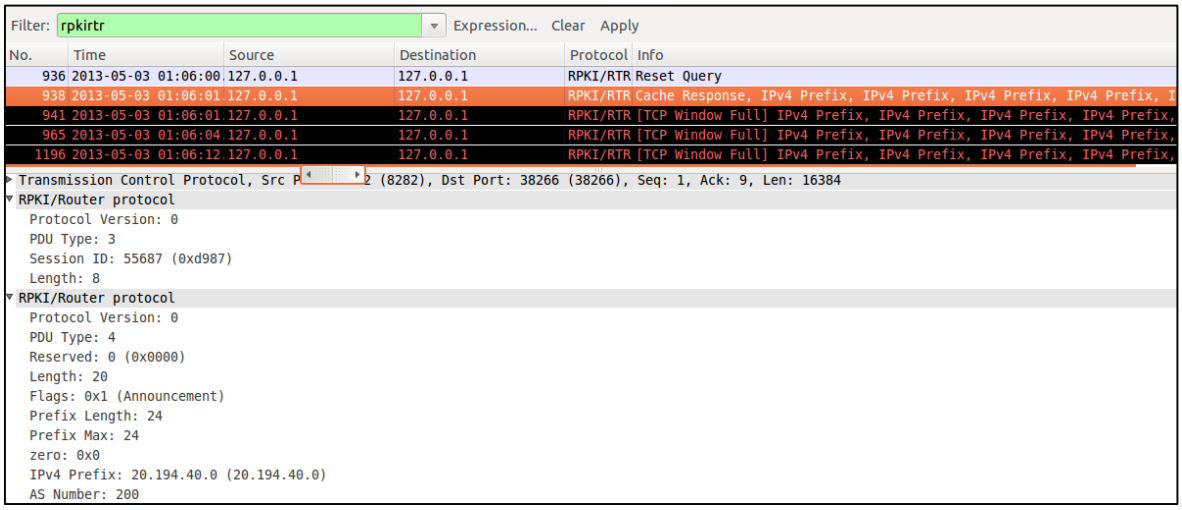


Obrázek 14: Komunikace mezi RPKI validátorem a Quaggou za pomocí SRxProxy serveru.

Pro zachytávání výměny informací, za použití SRX proxy serveru, mezi validátorem a samotnou Quaggou je poskytováno dvou zásuvných modulů pro aplikaci „Wireshark“ kterými jsou SRXproxy a RPKI/rtr“. Tyto dva moduly jsou k dispozici opět na výše zmiňovaném odkazu. Jak vyplývá už z předchozího obrázku, zásuvný modul SRXProxy nám monitoruje provoz mezi SRX serverem a QuaggouSRX a modul RPKI/rtr nám monitoruje provoz mezi validátorem a SRX serverem. Jednotlivé pakety zachycené programem wireshark jsou vidět na Obrázek **15** a Obrázek **16**



Obrázek 15: Zachycení paketů „SRXproxy“ protokolu pomocí Wireshark



Obrázek 16: Zachycení paketů RPKI/rtr protokolu pomocí Wireshark

4.6.2 QuaggaSRX

Jedná se o volně dostupnou implementaci směrovače Quagga s integrací SRX server API [24]. Nastavení tohoto směrovače je podobné jako u směrovače CISCO. Nastavení spojení se SRX serverem a nastavení chování procesu validace můžeme nastavit pomocí následujících příkazů.

4.6.2.1 Konfigurační příkazy pro QuaggaSRx

srx display – Zapnutí zobrazení SRx informací pro příkazy „show“

srx-proxy-id - SRx server používá tuto hodnotu k identifikaci směrovače. Hodnota srx-proxy-id musí být nastavena dříve, než se směrovač připojí k SRx serveru. Tuto hodnotu je dobré použít stejnou jako router-id.

srx set-server <host> <port> - nastavení adresy SRx serveru bez přímého připojení.

srx connect – připojení k SRx serveru. Při tomto příkazu je možné také nastavit <host> a <port> serveru.

srx disconected – odpojení od serveru

srx keep-window - čas (uvedený v sekundách) po který si SRx server uchovává informace. Po vypršení limitu jsou smazány. To umožňuje směrovač restartovat bez ztráty výsledků validace v rámci SRx.

srx evaluation origin_only – povolení zpracování politik v rámci rozhodovacího procesu.

no srx evaluation – zákaz zpracovávání politik v rámci rozhodovacího procesu. QuaggaSrx se přepne do normálního BGP zpracovávání a však stále si vyměňuje informace se serverem a přijímá oznámeni od tohoto serveru, to však nemá na zpracovávání BGP zpráv žádný vliv.

srx set-origin-value <valid>, <notfound>, <invalid>, <undefined> - nastavení výchozí hodnoty pro ověření původu

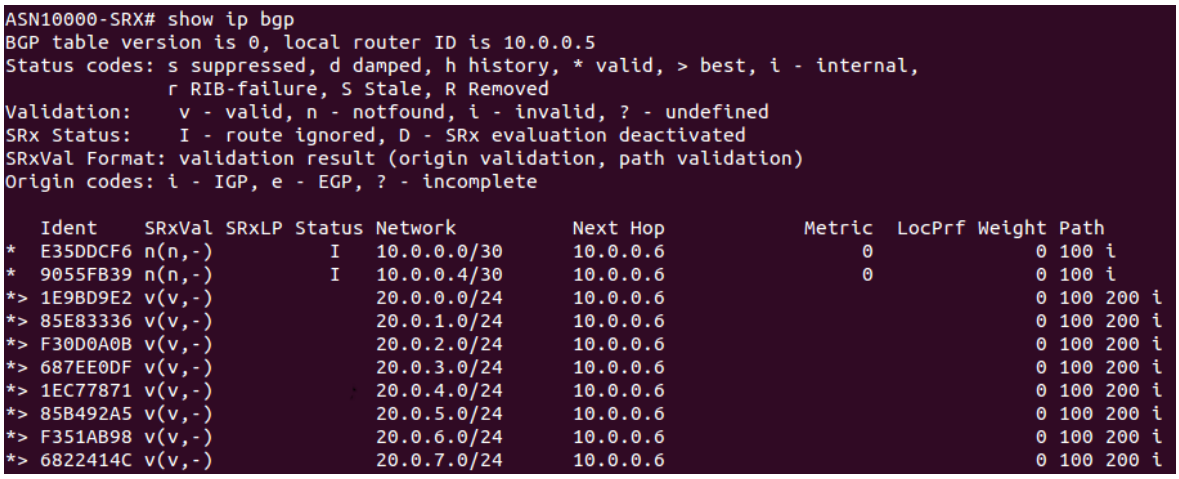
4.6.2.2 Nastavení politik

[no] srx policy (ignore-notfound | ignore-invalid | ignore-undefined) – nastavení politik pro ignorování jednotlivých zpráv Update, které jsou ve stavu notfound, invalid a undefined. Jednotlivé zprávy nebudou dále zpracovávány. Výchozí nastavení je však ignore-undefined. Síťě, které jsou označeny jako „undefined“ mají přechodný stav, dokud nedojde k validaci těchto sítí. Jakmile jsou tyto zprávy zpracovány, tak obdrží stav valid, notfound nebo invalid.

srx policy local-preference (valid | notfound | invalid) <value> [add | subtract] – nastavení politik umožňující zprávy se stavem valid, notfound, invalid přepsat nebo upravit fixní lokální preferenci, přičemž pro úpravy je potřebné přidat parametry „add“ nebo „subtract“. Úprava hodnoty local-preference umožnuje kombinovat ostatní politiky validace.

[no] srx policy prefer-valid – politika, která upřednostňuje validní zprávy před těmi, které jsou jiného stavu.

Na následujícím obrázku můžeme vidět funkčnost validace kdy jednotlivé sítě, které byly oznámeny z AS200 a pro které existovalo ROA, byly prohlášeny za validní a kde v hodnotě „SrxVal“ vidíme hodnotu v(v, -) a dále sítě které byli prohlášeny za notfound n(n, -) a v hodnotě „Status“ vidíme, že také zafungovali nastavení politik, které byly nastaveny a sítě, které nebyly validní, správně nebyly propagovány dále a byly úspěšně ignorovány.

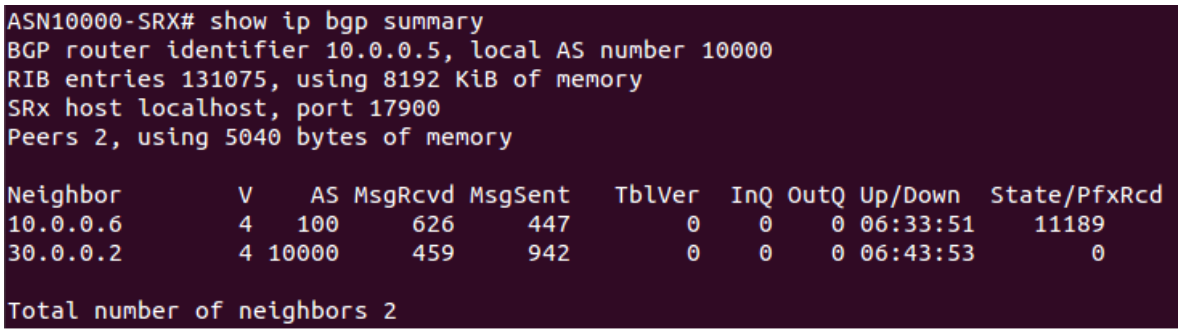


```
ASN10000-SRX# show ip bgp
BGP table version is 0, local router ID is 10.0.0.5
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal,
              r RIB-failure, S Stale, R Removed
Validation:    v - valid, n - notfound, i - invalid, ? - undefined
SRx Status:    I - route ignored, D - SRx evaluation deactivated
SRxVal Format: validation result (origin validation, path validation)
Origin codes: i - IGP, e - EGP, ? - incomplete

   Ident    SRxVal SRxLP Status Network          Next Hop            Metric  LocPrf Weight Path
*  E35DDCF6 n(n,-)          I   10.0.0.0/30      10.0.0.6                 0              0 100 i
*  9055FB39 n(n,-)          I   10.0.0.4/30      10.0.0.6                 0              0 100 i
*> 1E9BD9E2 v(v,-)              20.0.0.0/24      10.0.0.6                                0 100 200 i
*> 85E83336 v(v,-)              20.0.1.0/24      10.0.0.6                                0 100 200 i
*> F30D0A0B v(v,-)              20.0.2.0/24      10.0.0.6                                0 100 200 i
*> 687EE0DF v(v,-)              20.0.3.0/24      10.0.0.6                                0 100 200 i
*> 1EC77871 v(v,-)              20.0.4.0/24      10.0.0.6                                0 100 200 i
*> 85B492A5 v(v,-)              20.0.5.0/24      10.0.0.6                                0 100 200 i
*> F351AB98 v(v,-)              20.0.6.0/24      10.0.0.6                                0 100 200 i
*> 6822414C v(v,-)              20.0.7.0/24      10.0.0.6                                0 100 200 i
```

Obrázek 17: Výpis ověřených sítí pomocí příkazu „show ip bgp“

Pomocí příkazu „show ip bgp summary“ (Obrázek **18**) můžeme vidět 11189 ověřených a validních sítí, které byly ověřeny pomocí námi vytvořených ROA, které byly oznámeny od souseda 10.0.0.6. Toto nám dokazuje, že ostatní sítě, které byly oznámeny, byly úspěšně filtrovány a také nebyly dále propagovány.



```
ASN10000-SRX# show ip bgp summary
BGP router identifier 10.0.0.5, local AS number 10000
RIB entries 131075, using 8192 KiB of memory
SRx host localhost, port 17900
Peers 2, using 5040 bytes of memory

Neighbor        V    AS MsgRcvd MsgSent   TblVer  InQ OutQ Up/Down  State/PfxRcd
10.0.0.6        4   100     626     447        0    0    0 06:33:51    11189
30.0.0.2        4 10000     459     942        0    0    0 06:43:53        0

Total number of neighbors 2
```

Obrázek 18: Výpis sumarizovaných cest pomocí příkazu „show ip bgp summary“

Abychom zjistili stav konkrétních sítí, které jsou v BGP tabulce směrovače, použijeme příkaz „show ip bgp <network> kde můžeme zjistit informace o validaci. Tato informace se nachází v detailu SRx Information. Následující Obrázek **19** nám dokazuje, že pro síť 20.0.0.10/24 existovalo ROA, podle kterého se stala tato síť validní. (hodnota prefix-origin: valid)

Obrázek 19: Výpis ověřené sítě 20.0.0.1/24 pomocí příkazu „show ip bgp <network>“

4.7 Měření a porovnání rychlosti validátorů

U validace samotných stažených informací z CA byly pozorovány rozdíly v chování jednotlivých validátorů. Při zpracovávání objektů z našeho lokálního repositáře, RIPE NCC validátor byl při zpracovávání tohoto menšího počtu ROA pomalejší než validátor RCYNIC. V případě nastavení jednotlivým validátorům více cest k repositářům (pomocí souborů *.tal), ke kterým se mají jednotlivé validátory připojit, bylo zjištěno, že validátor RCYNIC je pomalejší, než RIPE NCC validátor. Toto je způsobeno tím, že RIPE NCC validátor stahuje a procesuje objekty všech repositářů zároveň. V případě validátoru RSYNC se procesují objekty z jednotlivých repositářů postupně. Validátor RIPE je o mnoho náročnější na alokovanou paměť. V obou případech validace, jak u RIPE tak i u RSYNC byl procesor vytížen mnohdy na plný výkon. To by mohlo znamenat, že rychlost procesu validace může být ovlivněn použitým hardwarem. Dobu zpracování procesu validace pomocí RIPE NCC RPKI a RCYNIC validátoru při zpracování objektů z lokálního repositáře můžeme vidět v následujících tabulkách.

Číslo měření	čas [mm:ss]
1	00:59
2	01:00
3	01:18
4	01:05
5	01:08
6	01:11
Průměr	**01:07**

Tabulka 1 - Rychlost procesu validace pomoci RIPE NCC RPKI validátoru

Číslo měření	čas [mm:ss]
1	00:21
2	00:21
3	00:26
4	00:20
5	00:29
6	00:21
Průměr	**00:23**

Tabulka 2 - Rychlost procesu validace pomoci RCYNIC validátoru

Naopak dobu zpracování procesu validace pomocí RIPE NCC RPKI a RCYNIC validátoru při zpracování objektů z více repositářů můžeme vidět v následujících tabulkách.

Číslo měření	čas [mm:ss]
1	05:20
2	04:49
3	04:31
4	04:45
5	06:15
6	05:01
Průměr	**05:07**

Tabulka 3 - Rychlost procesu validace pomoci RIPE NCC RPKI validátoru

Číslo měření	čas [mm:ss]
1	35:47
2	30:03
3	29:16
4	35:00
5	29:24
6	31:38
Průměr	**31:51**

Tabulka 4 - Rychlost procesu validace pomoci RCYNIC validátoru

Abychom mohli otestovat délku trvání propagace BGP sítí ze směrovače QuaggaPC3 do směrovače QuaggaPC4 musel jsem najít jakým způsobem toto změřit. Nejlepší způsob, jak tuto propagaci sítí nejlépe změřit byl za pomocí programu wireshark, s nímž se monitorovali jednotlivé časy příchozích zpráv UPDATE do směrovače QuaggaPC4. Čas byl měřen od spuštění směrovače QuaggaPC4. Sítě uvedené ve zprávách UPDATE byly před tím ověřeny a označeny směrovačem QuaggaSRx jako validní. Sítě, které nebyly označeny ve směrovači QuaggaSRx jako validní, nebyly dále kvůli použité politice dále propagovány. Rychlost propagace sítí bez ověřování byla jen chvilková což postrátalo smysl měřit. Proces

propagace sítí s validací trval v průměru 12 minut a 22 sekund. Zjistil jsem, že proces ověřování jednotlivých sítí se SRX serverem, vytíží procesor také mnohdy na plný výkon. (opět dokazuje nedostatek v použitém hardwaru) V následující tabulce vidíme čas ověření všech validních cest a rozšíření do ostatních směrovačů.

Číslo měření	čas [mm:ss]
1	13:00
2	13:30
3	12:34
4	11:18
5	12:07
6	11:44
Průměr	12:22

Tabulka 5 - Čas ověření všech validních cest a jejich rozšíření do směrovače QuaggaPC4

V případě, že se odpojí směrovač QuaggaPC3 začnou se z paměti SRx serveru odebírat informace o validaci. Dokud se neodeberou z paměti všechny informace je proces rozšíření všech sítí při obnovení směrovače QuaggaPC3 rychlejší.

Číslo měření	čas [mm:ss]
1	07:03
2	08:31
3	07:05
4	10:12
Průměr	**08:13**

Tabulka 6 - Rychlost odebrání informací o validaci

5 Závěr

V této diplomové práci jsem prozkoumal a zdokumentoval stávající stav implementací pro zabezpečení BGP infrastruktury pomocí S-BGP, soBGP a RPKI.

Při zpracovávání diplomové práce jsem vycházel převážně ze zahraničních zdrojů, ve kterých jsou jednotlivé problematiky popsány. Většinou se jednalo o RFC specifikace, které poskytovaly detailnější popis této rozsáhlé problematiky, který vedl k lepšímu pochopení jednotlivých nalezených architektur. Všechny znalosti, které mi poskytly, již zmiňované dokumenty, mi pomohly vypracovat praktickou část této práce, která obsahuje nasazení implementace certifikační autority, RIPE NCC a RCYNIC validátoru a směrovače QuaggaSRX pro otestování infrastruktury RPKI. Tyto testovací implementace jsou zcela funkční a k dispozici na přiložených DVD. V práci jsem také uvedl jednotlivé síťové prvky, které jsou pro nasazení RPKI architektury připraveny a v dnešní době také poskytovány v omezené míře i pro testovací účely organizací RIPE.

V dnešní době se stále rostoucími riziky zabezpečení systémů a dat proudících do těchto systémů, je zapotřebí v neposlední řadě mít stále více zabezpečenou i síťovou infrastrukturu. Dalším hlavním cílem pro zabezpečení této problematiky je v dnešní době stále vyvíjející se architektura BGPSEC, která je založena na tom, jak opatřit digitálním podpisem celou BGP cestu od zdroje až k cíli. Toto však na reálné nasazení si bude muset ještě nějakou dobu počkat, protože nasazení BGPSEC do produkčního prostředí se plánuje na rok 2016 [25].

Na závěr lze říci, že tato diplomová práce shrnuje jednotlivé myšlenky zabezpečení BGP infrastruktury. Ať už jsou mnohé protokoly používané pro zabezpečení BGP infrastruktury pozapomenuty, jsou stále dokumentovány nové myšlenky pro docílení co nejlepšího zabezpečení, které neklade tak velké paměťové zatížení na samotné síťové komponenty a architektura RPKI je toho také příkladem.

6 Literatura

[1] Lynn Ch., Mikkelson J., Karen S. Internet Draft: *Secure BGP (S-BGP)*, 2003. http://tools.ietf.org/html/draft-clynn-s-bgp-protocol-01

[2] Kent S.T., Lynn Ch., Mikkelson J., Seo K. *Secure Border Gateway Protocol (S-BGP) Real World Performance and Deployment Issue,* 2000. http://users.ece.cmu.edu/~adrian/731-sp04/readings/KLMS-SBGP.pdf

[3] Weis B. Internet-draft: *Secure Origin BGP (soBGP) Certificates*. 2006 http://tools.ietf.org/html/draft-weis-sobgp-certificates-04

[4] Russ W. Internet-draft: *Architecture and Deployment Considerations for Secure Origin BGP (soBGP), 2004.* http://tools.ietf.org/html/draft-white-sobgparchitecture-00

[5] Ootschot P.C., Wan T.and Kranakis E. On Interdomain Routing Security and Pretty Secure BGP (psBGP).Verze 11.6.2007. http://people.scs.carleton.ca/~paulv/papers/tissec-july07.pdf

[6] Cooper D., Santesson S., Ferrell S., Boeyen S., Housley R., Polk W.: *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*. RFC5280. 2008. http://tools.ietf.org/html/rfc5280

[7] Lynn C., Kent S., Seo K.: *X.509 Extensions for IP Addresses and AS Identifiers*. RFC3779, 2004. http://tools.ietf.org/html/rfc3779

[8] Lepinski M., Chi A. and Kent S.: *Signed Object Template for the Resource Public Key Infrastructure (RPKI)*. RFC6488, Internet Engineering Task Force (IETF), 2012, ISSN: 2070-1721. http://tools.ietf.org/html/rfc6488

[9] Lepinski M. and Kent S.: *An Infrastructure to Support Secure Internet Routing*. RFC6480, Internet Engineering Task Force (IETF), 2012, ISSN: 2070-1721. http://tools.ietf.org/html/rfc6480

[10] Huston G., Michaelson G. and Loomans R: *A Profile for X.509 PKIX Resource Certificates*. RFC6487, Internet Engineering Task Force (IETF), 2012, ISSN: 2070-1721. http://tools.ietf.org/html/rfc6487

[11] Huston G.: *The Profile for Algorithms and Key Sizes for Use in the Resource Public Key Infrastructure (RPKI)*. RFC6485, Internet Engineering Task Force (IETF), 2012, ISSN: 2070-1721. http://tools.ietf.org/html/rfc6485

[12] Huston G., Weiler S., Michaelson G. and Kent S.: *Resource Public Key Infrastructure (RPKI) Trust Anchor Locator*. RFC6490, Internet Engineering Task Force (IETF), 2012, ISSN: 2070-1721. http://tools.ietf.org/html/rfc6490

[13] Austein R., Huston G., Kent S. and Lepinski M.: *Manifests for the Resource Public Key Infrastructure (RPKI)*. RFC6486, Internet Engineering Task Force (IETF), 2012, ISSN: 2070-1721. http://tools.ietf.org/html/rfc6486

[14] Lepinski M., Kent S. and Kong D.: *A Profile for Route Origin Authorizations (ROAs)*. RFC6482, Internet Engineering Task Force (IETF), 2012, ISSN: 2070-1721. http://tools.ietf.org/html/rfc6482

[15] Huston G. Michaelson G.: *Validation of Route Origination Using the Resource Certificate Public Key Infrastructure (PKI) and Route Origin Authorizations (ROAs)*. RFC6483, Internet Engineering Task Force (IETF), 2012, ISSN: 2070-1721. http://tools.ietf.org/html/rfc6483

[16] Huston G., Loomans R. and Michaelson G.: *A Profile for Resource Certificate Repository Structure*. RFC6481, Internet Engineering Task Force (IETF), 2012, ISSN: 2070-1721 http://tools.ietf.org/html/rfc6481

[17] Bush R., Austein R.: *The Resource Public Key Infrastructure (RPKI) to Router Protocol.* RFC6810, Internet Engineering Task Force (IETF), 2013. ISSN: 2070-1721. http://tools.ietf.org/html/rfc6810

[18] ARIN. rpki.net project site. Dostupé na World Wide Web: https://trac.rpki.net/.

[19] RIPE validation tool: RIPE NCC RPKI Validator. Dostupné na Word Wide Web: https://www.ripe.net/lir-services/resource-management/certification/tools-and-resources

[20] BBN Validation Tool: *The Relying Party Security Technology for Internet Routing (RPSTIR)*. Dostupné na Word Wide Web: http://sourceforge.net/projects/rpstir/

[21] National Institute of Standards and Technology. *BGP Secure Routing Extension (BGP-SRx)*. Verze duben 2013. http://www-x.antd.nist.gov/bgpsrx/

[22] Borchert O., NIST. *Secure Routing Extension (SRx) - Proxy Protocol*. http://www-x.antd.nist.gov/bgpsrx/documents/srx-server-protocol-1-0.txt.pdf

[23] Borchert O., Lee K., Sriram K. and Montgomery T. *BGP Secure Routing Extension – BGP- SRx*. Verze 0.3 Březen 2013. http://www-x.antd.nist.gov/bgpsrx/documents/BGPSRxUsersManual-3.pdf

[24] Borchert O., Lee K., Sriram K. and Montgomery T. *Quagga and BGP Secure Routing Extension – QuaggaSRx*. Verze 0.3 Březen 2013. http://www-x.antd.nist.gov/bgpsrx/documents/QuaggaSRxUsersManual-3.pdf

[25] Sriram K., Borchert O., Kim O., Cooper D., Montgomery D. *RIB Size Estimation for BGPSEC*. Dostupné na Word Wide Web: http://www.nist.gov/itl/antd/upload/BGPSEC_RIB_Estimation.pdf

A Obsah DVD

Následující tabulky popisují umístění souborů na DVD a stručný popis.

DVD1 - adresář	Popis
VMWARE – CA, BGP-SRX, Validators	VMWARE image pro PC1, kde je nasazena CA, SRX server směrovač QuaggaSRX, RCYNIC a RIPE NCC validátor.

DVD2 - adresáře	Popis
VMWARE - Quagga PC2	VMWARE image pro Quagga PC2 s Quagga smerovačem
VMWARE - Quagga PC3	VMWARE image pro Quagga PC2 s Quagga smerovačem

DVD3 - adresáře	Popis
VMWARE - Quagga PC3	VMWARE image pro Quagga PC4 s Quagga smerovačem
Dokument	Diplomová práce ve formátu PDF
HKS	Hlavní konfigurační soubory